Circuit Techniques for Low-Power and Secure Internet-of-Things Systems

by

Kaiyuan Yang

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctorate of Philosophy
(Electrical Engineering)
in The University of Michigan
2017

Doctoral Committee:

Professor Dennis M. Sylvester, Chair
Professor Todd M. Austin
Professor David T. Blaauw
Associate Professor Kevin P. Pipe

Kaiyuan Yang

kaiyuan@umich.edu

ORCID id: 0000-0001-7220-9389

To My Family

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

Table

ABSTRACT

The coming of Internet of Things (IoT) is expected to connect the physical world to the cyber world through ubiquitous sensors, actuators and computers. The nature of these applications demand long battery life and strong data security. To connect billions of things in the world, the hardware platform for IoT systems must be optimized towards low power consumption, high energy efficiency and low cost. With these constraints, the security of IoT systems become a even more difficult problem compared to that of computer systems. A new holistic system design considering both hardware and software implementations is demanded to face these new challenges.

In this work, highly robust and low-cost true random number generators (TRNGs) and physically unclonable functions (PUFs) are designed and implemented as security primitives for secret key management in IoT systems. They provide three critical functions for cryptographic systems including runtime secret key generation, secure key storage and lightweight device authentication. To achieve robustness and simplicity, the concept of frequency collapse in multi-mode oscillator is proposed, which can effectively amplify the desired random variable in CMOS devices (i.e. process variation or noise) and provide a runtime monitor of the output quality. A TRNG with self-tuning loop to achieve robust operation across -40 to 120 degree Celsius and 0.6 to 1V variations, a TRNG that can be fully synthesized with only standard cells and commercial placement and routing tools, and a PUF with runtime filtering to achieve robust authentication, are designed based upon this concept and verified in several CMOS technology nodes. In addition, a 2-transistor sub-threshold amplifier based "weak" PUF is also presented for chip identification and key storage. This PUF achieves state-of-the-art 1.65% native unstable bit, 1.5fJ per bit energy efficiency, and 3.16% flipping bits across -40 to 120 degree Celsius range at the same time, while occupying only 553 feature size square area in 180nm CMOS.

Secondly, the potential security threats of hardware Trojan is investigated and a new Trojan

attack using analog behavior of digital processors is proposed as the first stealthy and controllable fabrication-time hardware attack. Hardware Trojan is an emerging concern about globalization of semiconductor supply chain, which can result in catastrophic attacks that are extremely difficult to find and protect against. Hardware Trojans proposed in previous works are based on either design-time code injection to hardware description language or fabrication-time modification of processing steps. There have been defenses developed for both types of attacks. A third type of attack that combines the benefits of logical stealthy and controllability in design-time attacks and physical ”invisibility” is proposed in this work that crosses the analog and digital domains. The attack eludes activation by a diverse set of benchmarks and evades known defenses.

Lastly, in addition to security-related circuits, physical sensors are also studied as fundamental building blocks of IoT systems in this work. Temperature sensing is one of the most desired functions for a wide range of IoT applications. A sub-threshold oscillator based digital temperature sensor utilizing the exponential temperature dependence of sub-threshold current is proposed and implemented. In 180nm CMOS, it achieves 0.22/0.19K inaccuracy and 73mK noise-limited resolution with only 8865 square micrometer additional area and 75nW extra power consumption to an existing IoT system.

CHAPTER 1

Introduction

1.1 Emerging Internet-of-Things Applications

Rapid developments of semiconductor industry over the last few decades have enabled technologies that were once considered far-fetched imaginations. For example, the development of personal computers opened the path towards a digital world, where information can be shared instantly and numerous life-changing services have been created. The coming of smart mobile devices further strengthen the trend and totally reshaped our daily life. This trend of shrinking electronic devices is predicted by by Bell's Law [1], as shown in Figure 1.1. This trend continues and the next generation of ubiquitous devices can be envisioned, which will internetwork our physical world to the cyber world and make everything smarter. Such a system is usually called the Internet of Things (IoT) or Internet of Everything (IoE). IoT will enable numerous life-changing applications, such as smart city, smart home, securely connected autonomous cars, and implantable/wearable medical devices.

The fundamental technology enablers of such systems are ubiquitous devices equipped with sensors, actuators, computers, and network connectivity that enable the physical things to collect and exchange data. Low-power and low-cost integrated circuits are foundations for hardware platforms of these devices. Nowadays, the need for better circuit designs with CMOS and post-CMOS devices is stronger than ever because we can no longer simply take advantage of CMOS technology scaling, as Moores law drawing to an end in near future. However, power and cost are not the only critical factors to consider. Since all the aforementioned applications demand tremen-

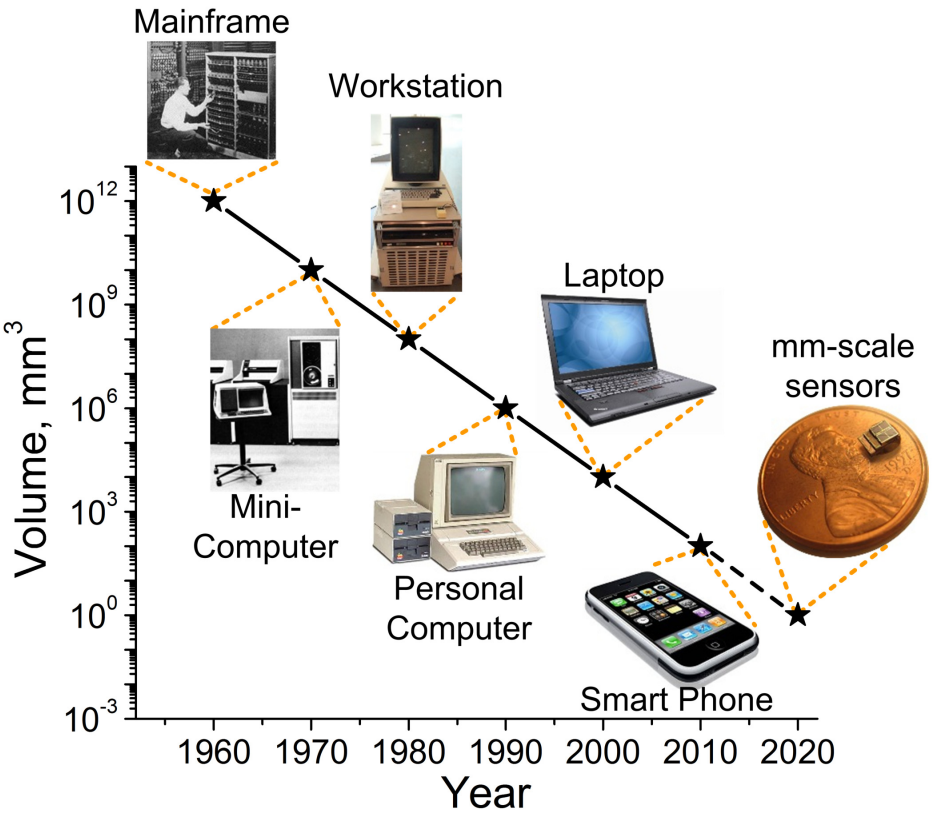


Figure 1.1: Bell's Law on scaling of computing platforms.

dous amount of devices to be connected and privacy/safety data to be processed and transmitted, security and privacy issues must be solved before IoT-led technology revolutions become reality. With so many alarming stories about attacks to commercial IoT systems nowadays (e.g. massive distributed denial-of-service (DDoS) attack to US east coast caused by insecure IoT web cameras in 2016 [2], carjacking of Jeep on highway [3] and the fear of former US Vice President Cheney's heart defibrillator being hacked [4]), security is becoming a fundamental requirement rather than simply additional features. Securing IoT systems faces additional challenges beyond conventional computer system and Internet security due to limited computing resources of the devices, stringent power budgets, severe cost pressures, and physical accessibility of these devices to attackers. At the lowest level of the system stack, hardware security building blocks and defenses, are critical components to support the security of the complete system.

1.2 Secure Key Management in Silicon

Secret key is the most critical data to protect in most secure systems, because once the key is stolen or broken by attackers with any software or hardware attacks, the whole cryptographic system will be corrupted. Secret keys are used in a wide range of cryptographic primitives including secret (symmetric) key cipher for confidential communication, public (asymmetric) key cipher for data authentication, and secure hash functions for message integrity. The two basic functions for secret key management are run-time secret key generation and secure key storage for pre-installed keys.

For secret key generation, true random number generator (TRNG) harvesting entropy from physical random sources is preferred over pseudo random number generator (PRNG) based on mathematic functions and an initial "seed". In computer systems, the "seed" usually comes from "random" user input or system events. But for IoT applications, relying on external inputs or system events is impractical and dangerous. The quality of generated true random numbers are verified by statistical testings, out of which NIST 800-22 random test suite [5] is the most comprehensive and strict test. In addition to randomness, robustness of TRNG to process, voltage and temperature (PVT) variations, resistance to intentional attacks (e.g. power attack), area, throughput and energy efficiency are other metrics to consider.

For secure key storage, conventional method of using non-volatile memories (NVM) suffers from a few issues, such as re-programming, data retention and direct optical reading. Moreover, NVMs mostly require extra processing steps and costs, which are not desirable for IoT systems. Physically unclonable function (PUF) has emerged as a potential solution to the secret key storage challenge. PUFs extract entropy from hardware intrinsic process variations. The responses are unique to each fabricated chip and therefore can be used as random and unique IDs and keys for authentication and encryption. PUFs also have the promise of lightweight and secure device authentication for IoT device communication and semiconductor supply chain protection. Uniqueness and reproducibility are the two fundamental requirements for PUFs, which can be measured by inter-PUF and intra-PUF Hamming Distances (see Section 4.3 for detailed explanations).

To design TRNGs and PUFs, available random sources in CMOS technologies include device variations and noise. Device variation shows an approximately normal distribution of threshold

voltages across fabricated devices while noise is changing over time and independent for different devices. Therefore, TRNG designs should maximize noise level to harvest entropy, while PUF designs should amplify device variations for uniqueness and suppress noise for reproducible readings. Within the contexts of IoT systems, these designs require robustness across wide environmental (voltage and temperature) conditions, compact design, technology portability and low power consumption.

1.3 Malicious Hardware Attack

There have been a wide range of attacks making using of software-level vulnerabilities and malicious Trojans. As most defense efforts also happen at software stacks, it can be expected that more attacks will be targeting lower stacks of a computer (i.e. hardware) that do not have much protections nowadays. Malicious hardware, or hardware Trojan, is one of such attacks that raises a lot of concerns. This attack aims at inserting back-doors in integrated circuits, which waits for certain triggering conditions to activate attacks that can bypass higher-level security protections. The threat of these attacks is accompanied by the globalization of semiconductor supply chain. Due to huge investment in the semiconductor industry, it has been divided into many sectors spread all over the world.

Both design-time and fabrication-time attacks have been described in literature. Design-time attack happens in 3rd party IPs or RTL designs, which means the attack exists in logical abstraction layer. Previous fabrication-time attacks targets to directly modify device parameters to cause malfunctions. There have been many defense efforts in literature as well, mostly fall in two categories: visual or side-channel inspection; and static and dynamic logic analysis.

To better protect against these attacks, it is important to consider both defenses to previous attacks presented and potential new attacks that thwart existing defenses. In this work, a new fabrication-time attack that possesses the advantages of previous design-time and fabrication-time attacks is introduced in Chapter 6. The attack, called A2, exploits the analog behavior of digital processors to trigger the attack. It can evade existing defenses and deploy powerful attacks to the system.

1.4 Low-power Temperature Sensing

Temperature sensing is one of the most desired functions for many IoT applications in medical, industrial, automotive and consumer fields. To enable long-term monitoring in IoT systems using batteries, the peak power consumption and energy per conversion should both be minimized for the sensors. At the same time, severe cost constraints limit the types of devices, silicon area and number of trimming points (i.e. the number of different temperatures the sensor must be tested for calibration).

The performance of temperature sensors is measured by inaccuracy (linearity), resolution, conversion time, and energy efficiency. First of all, inaccuracy measures the maximum linearity error between sensor reading and real temperature across a defined operation range. Long enough conversion time and averaging over many readings are used to filter out noise effects. Two metrics exist in literature: (1)3σ (standard deviation) value of the max error of each device over enough samples; or (2) the maximum error across all samples. Secondly, resolution is the effective minimum step of sensor readings, which is usually limited by noise, but can also be limited by quantization if each step is too large. Noise-limited resolution is measured by the standard deviation of many consecutive sensor readings of same device and at constant temperature. Thirdly, conversion time is the time it takes the sensor to provide one reading. The conversion time is usually directly related to the noise-limited resolution. Longer conversion time provides more noise averaging and therefore better resolution, at the cost of higher energy consumption. The total energy it takes a sensor to do a conversion is the energy per conversion. It should be noted that certain sensor structures (e.g. $\Delta\Sigma$-ADC) provide better conversion time with less energy requires longer setup time, which must be considered for low-power and heavy duty-cycled IoT systems. In summary, a trade-off between these parameters should be achieved through novel sensor designs, and ideally, should be optimized considering system architectures.

1.5 Contributions and Organization of this Work

The contributions of this work to developing future low-power and secure IoT devices include novel security primitives designs (true random number generators and physically unclonable func-

tions) for secret key management, a low-power and high accuracy temperature sensor and revealing the possibility of hardware Trojan attack using analog behaviors of digital circuits.

In Chapter 2, I present an all-digital Edge Racing TRNG based on the collapse time of two chasing edges in an even-stage ring oscillator (RO) with automatic tuning loop, demonstrating extensive robustness against PVT variations and intentional power supply attacks [6]. The usage of oscillation collapse time in even-stage RO provide three benefits. (1) Easy detection of collapse event: no phase detector is needed and thus there are less non-ideal effects involved; (2) Average collapse time is naturally an indicator of the operation condition of the TRNG, on which the automatic tuning loop is based. (3) Tuning does not introduce bias into output bits and a relatively wide target range is acceptable; this eliminates the need for high resolution tuning, minimizing design complexity and cost. The TRNG has been fabricated in 40nm CMOS demonstrating 2Mb/s and 23pJ/b at nominal 0.9V while passing all 15 NIST randomness tests across wide operating conditions (-40 to 120$^{\circ}C$ and 0.6 to 0.9V). A second prototype in 180nm demonstrates its portability to an older technology commonly used for ultra low power applications like sensor nodes.

In Chapter 3, a fully synthesized true random number generator (TRNG) from standard cells is described, which can pass all NIST randomness tests. The design is fabricated in both 28nm and 65nm CMOS, demonstrating ease of technology portability. The TRNG harvests entropy from the time it takes an oscillator initialized to its 3rd harmonics to return to its fundamental frequency. Since all three edges in the oscillator go through exactly delay cells, process variation of delay cells can be canceled. Thus, the TRNG requires no pre-calibration or post-processing to achieve good randomness. At 23.16Mb/s, it consumes 0.54mW and occupies 375μm^2 in 28nm. Tolerance to power supply injection attack is achieved using a noise filter and verified by a built in test structure.

In Chapter 4, I describe a physically unclonable function (PUF) based on sub-threshold 2-transistor amplifier implemented in 180nm for secret key storage in IoT applications. Older technologies are preferred by ultra-low-power systems but present challenges to PUF designers because of less device mismatch. The proposed PUF cell with local amplification has a small footprint (553F^2), achieves 0.05% instability after 11-bit temporal majority voting, and 3.16%/2.01% flipping bits over -40 120$^{\circ}C$ and 0.8 1.8V. Hamming Distance is nearly ideal, with ¿700x separation between inter and intra Hamming Distances. The PUF runs at 4.8Gbps through wide paralleliza-

tion in an crossbar array, consuming 11.3 fJ/bit at 1.2 V and down to 1.5 fJ/bit at 0.8V.

In Chapter 5, an all-digital PUF with more than 5.510^28 challenge/response pairs for chip authentication is designed in 40nm CMOS. The design is based on frequency collapse in an even-stage ring oscillator. Compared to conventional arbiter PUF using delay chains, the proposed design can average out thermal noise for better reproducibility and provide a direct indicator of the confidence of the current output. With the help of dynamic thresholding, complementary-to-absolute-temperature (CTAT) bias generator and current starved delay cells, a BER of 0 is maintained across operating conditions from -25°C to 125°C and 0.7V to 1.2V in measurements. Average inter-chip hamming distance is nearly ideal 0.5007. The PUF has an effective throughput of 1.6Mb/s and occupies an area of $867\mu m^2$.

In Chapter 6, we design and implement the first fabrication-time processor attack that mimics the triggered attacks often added during design time. As a part of our implementation, we are the first to show how a fabrication-time attacker can leverage the empty space common to Application-Specific Integrated Circuit (ASIC) layouts to implement malicious circuits. It is shown that an analog attack can be much smaller and more stealthy than its digital counterpart. The attack diverts charge from unlikely signal transitions to implement its trigger, thus, it is invisible to all known side-channel defenses. Additionally, as an analog circuit, the proposed attack is below the digital layer and missed by functional verification performed on the hardware description language. Moreover, the attack relies on a complex and unlikely analog trigger sequence, thus, it is impractical to simulate at the analog level. To fully evaluate the attack, we fabricate a full processor with the proposed Trojan inserted and verifies that the Trojan can be activated by malicious programs while not exposed by normal embedded system test-benches. In addition, the behavior of our fabricated Trojans are characterized across many chips and environmental conditions.

In Chapter 7, a timing-based temperature sensor is designed specifically for IoT devices. An accurate temperature sensor usually requires a carefully calibrated, high-accuracy analog to digital converter, which prevents its use in ultra-low-power IoT sensor nodes. Observing that accurate timing sources are always available in IoT systems as a real-time clock (RTC) for time synchronization and time stamping, we proposed building a digital sensor utilizing the RTC as a reference to minimize the sensors power/area costs while achieving state-of-the-art performance (inaccuracy, resolution, and efficiency). Conventionally, timing-based temperature sensors suffer from

lower linearity and bad line sensitivity. To solve this, a new fitting method derived from device models is proposed to improve its linearity, a new differential delay cell is designed to provide a more linear conversion from sub-threshold current to oscillation frequency, a native NMOS header is used to regulate a local sub-threshold VDD, and a new counting scheme is proposed for such an oscillator exponentially dependent on temperature.

Finally, Chapter 8 concludes the completed work and discusses future directions in the field.

CHAPTER 2

Robust All-Digital True Random Number Generator

2.1 Introduction

For higher security level in most cryptographic systems, true random number generator (TRNG) harvesting entropy from physical sources are preferred over pseudo random number generator (PRNG) that has a fixed pattern decided by the seed. On-chip TRNG is important for system miniaturization and device noise provides a good entropy source for circuit designers. There have been a variety of designs extracting random number from device noise in literature. The conventional method is to amplify noise directly with a high gain and high bandwidth amplifier followed by quantization [7–9]. Resistor thermal noise [7], oxide trap noise [8] and SiN device noise [9] have been employed as entropy sources in this scheme. These designs require careful calibrations of the amplifier and ADC to remove bias in generated random numbers. Extensive use of analog designs also make them less attractive in terms of system integration and technology portability.

Digital TRNGs offer the advantages of easy integration and lower sensitivity to process, voltage and temperature variations (PVT variation) over conventional analog designs [10]. For mobile and IoT applications, robustness to environmental variations becomes even more critical. Previous works have demonstrated digital TRNGs based on metastability [10, 11], oscillator jitter [12–16], and other device noise (e.g., time to oxide breakdown [17]).Metastability-based methods using cross coupled inverters provide excellent operating frequency and power efficiency but often require extensive design efforts and run-time calibration to remove systematic and temporal mismatch in devices which are sensitive to environmental variations [10, 11]. A soft oxide breakdown

based TRNG provides high entropy random bits but suffers from low performance and low power efficiency due to the nature of the entropy source [17]. Ring oscillator (RO) jitter based TRNGs offer design simplicity and portability. Conventional methods using a slow RO to sample a jittery fast RO provide relative low entropy and low performance due to limited jitter in a single digital RO [12]. This design is also vulnerable to power supply attacks as described in [18]. Efforts to increase entropy of RO based TRNG include combining outputs of several parallel ROs [13], chaotic ROs with multiple feedback paths (FIRO and GARO) [19] and including a dynamic duty cycle tuning loop to remove bias in output [16]. Recent RO based TRNGs employ new random bit extraction schemes like measuring time for a 3rd harmonic RO to collapse to fundamental frequency [14] and beat frequency between 2 ROs running at close frequencies [15]. These new schemes provide better randomness and performance thanks to the new jitter amplification approaches, but robustness was not verified across PVT conditions and could pose difficulties to their applications.

To alleviate the issues of PVT variations, this work presents an all-digital Edge Racing TRNG based on the collapse time of two racing edges in an even-stage RO with automatic tuning loop, demonstrating extensive robustness against PVT variations and intentional power supply attacks [6]. The usage of oscillation collapse time in even-stage RO provide three benefits. (1) Easy detection of collapse event: no phase detector is needed and thus there are less non-idealities; (2) Average collapse time is naturally an indicator of the operation condition of the TRNG, on which the automatic tuning loop is based. (3) Tuning does not introduce bias into output bits and a relatively wide target range is acceptable; this eliminates the need for high resolution tuning, minimizing design complexity and cost. The TRNG has been fabricated in 40nm CMOS demonstrating 2Mb/s and 23pJ/b at nominal 0.9V while passing all 15 NIST randomness tests across wide operating conditions (-40 to 120$^{\circ}C$ and 0.6 to 0.9V). A second prototype in 180nm demonstrates its portability to an older technology commonly used for ultra low power applications like sensor nodes.

The remainder of the chapter is organized as follows. The concept of using frequency collapse in an even-stage RO as entropy source for true random number generation is described in Section 2.2; a mathematical model of the entropy source is also provided in Section 2.2; detailed implementation of the TRNG prototype and automatic tuning loop against PVT variations are described in Section 2.3; measurement results of both 40nm and 180 nm test chips are provided in Section 2.4; and finally, a summary is presented in Section 2.5.

2.2 Frequency Collapse Based TRNG

2.2.1 Analytical Model of Frequency Collapse in Even-stage RO

The main concept of the all-digital PVT tolerant Edge Racing TRNG is using the frequency collapse time in an even stage RO Fig 2.1 as entropy source. Two edges (A, B) are injected through NAND gates into opposite nodes of an even-stage RO simultaneously. Because of even number of stages, "A" is always rising at OUT port while"B" is always falling. For CMOS inverters, rising delay and falling delay are separated and can be changed by process variations. As shown by the arrows in Fig 2.1, the two injected edges travel entirely different paths through the ring. Taking device mismatch and random noise into consideration, the time for 2 edges to travel around the ring are separate accumulations of ideal delay, delay mismatch and noise. The time points of Nth rising and falling edges at OUT port of a RO with S stages can be expressed as,

$$T_{fall,N} = D_1 + \sum_{n=1}^{N}\sum_{i=1}^{S}(Ideal\_Delay_{i,B} + \Delta Delay_{i,B} + Jitter_{n,i,B}) \tag{2.1}$$

$$T_{rise,N} = D_2 + \sum_{n=1}^{N}\sum_{i=1}^{S}(Ideal\_Delay_{i,A} + \Delta Delay_{i,A} + Jitter_{n,i,A}) \tag{2.2}$$

where D_1 and D_2 are the time for edge "B" and "A" to reach OUT after start (Figure 2.1); $Ideal\_Delay_{i,B}$ and $Ideal\_Delay_{i,A}$ are the ideal delay of stage i for edge "B" and "A" not considering process variation and noise; $\Delta Delay_{i,A}$ and $\Delta Delay_{i,B}$ are the delay differences in addition to ideal delay due to process variation at stage i for corresponding edge; $Jitter_{n,i,A}$ and $Jitter_{n,i,B}$ represent the random delay caused by device noise at stage i during nth iteration of corresponding edge. Jitter is usually modeled as a random variable following normal distribution $N(0,\sigma^2)$. As modeled in [20], the variance of inverter delay due to white noise can be expressed as,

$$\sigma^2 = \frac{4kT\gamma_N t_{dN}}{I_N(V_{DD}-V_t)} + \frac{kTC}{{I_N}^2} \tag{2.3}$$

where t_{dN} is the window that noise is integrated during output transition, I_N is charging/discharging current, V_t is threshold voltage, γ is a technology dependent noise coefficient, C is loading capacitor of the inverter and k is Boltzmann constant. The last two terms in Equation 2.1 and

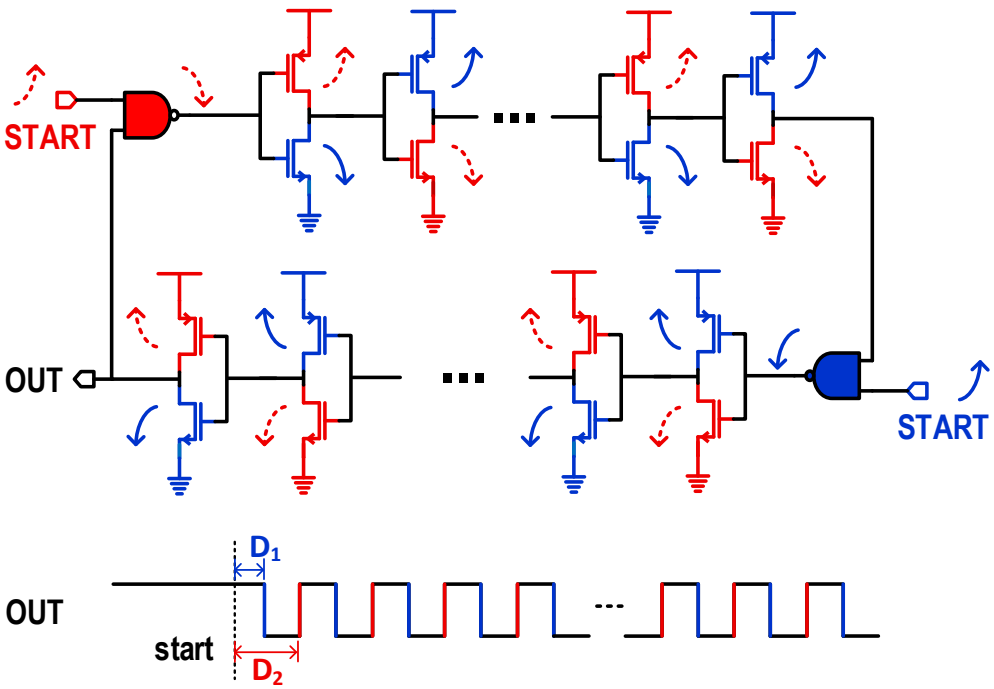


Figure 2.1: Concept of TRNG based on frequency collapse of edge racing RO.

Equation 2.2 represent non-idealities of the RO and cause one edge to travel faster than the other, thus overtaking the other and collapsing the oscillation. There are two possible collapse conditions depending on the relative amount of delay added to the 2 edges by process variation (Figure 2.2), which can be written as,

$$T_{rise,N} = T_{fall,N}, \text{if A is faster than B} \tag{2.4}$$

$$T_{rise,N} = T_{fall,N+1}, \text{if B is faster than A} \tag{2.5}$$

Substitute Equation 2.1 and 2.2 into Equation 2.4 and 2.5 and considering the fact that $\sum_{i=1}^{S} Ideal\_Delay_{i,A}$ and $\sum_{i=1}^{S} Ideal\_Delay_{i,B}$ are identical, the collapse conditions can be expressed as,

$$D_2 - D_1 = N \times \sum_{i=1}^{S} (\Delta Delay_{i,B} - \Delta Delay_{i,A}) + \sum_{n=1}^{N} \sum_{i=1}^{S} (Jitter_{n,i,B} - Jitter_{n,i,A}) \tag{2.6}$$

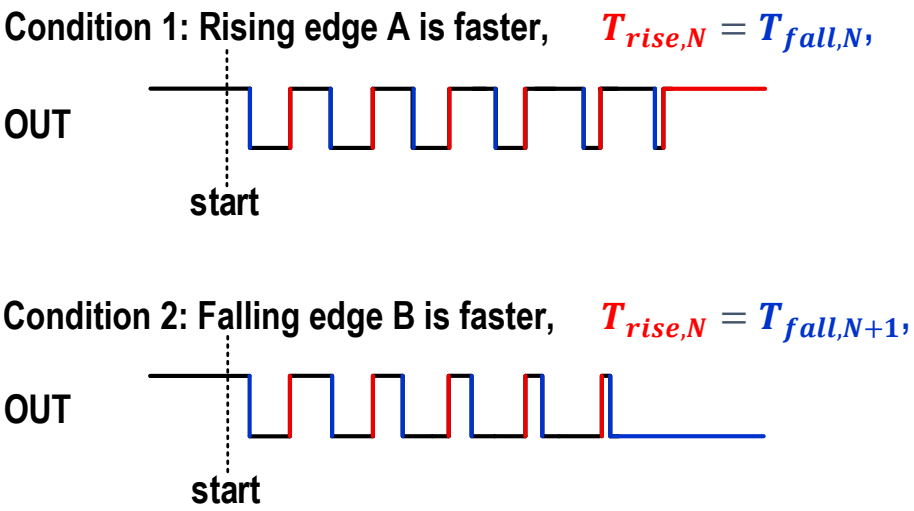


Figure 2.2: Even-stage RO collapse conditions and waveforms.

$$\begin{aligned} &D_2 - D_1 - \sum_{i=1}^{S}(\Delta Delay_{i,B}) - \sum_{i=1}^{S}(Jitter_{N+1,i,B}) \\ &\quad = N \times \sum_{i=1}^{S}(\Delta Delay_{i,B} - \Delta Delay_{i,A}) + \sum_{n=1}^{N}\sum_{i=1}^{S}(Jitter_{n,i,B} - Jitter_{n,i,A}) \end{aligned} \tag{2.7}$$

In Equation 2.6 and 2.7, left sides are constant for a given run, the first term on the right side is linear with number of cycles while the second term is accumulation of a normally distributed random variable. The right side can be viewed as a Gaussian random walk with drift in probability theory and therefore the model of the collapse event becomes the first-hitting-time model. As a result, the first-hitting-time (collapse time in our case) follows inverse Gaussian distribution [21]. Mean and variance of the time can therefore be derived from the model. Results of both conditions can be expressed in a unified form using absolute values, as shown below,

$$mean = \frac{|D_2 - D_1|}{|\sum_{i=1}^{S}(\Delta Delay_{i,B} - \Delta Delay_{i,A})} \tag{2.8}$$

$$variance = \frac{mean^3}{\lambda}, \lambda = \frac{(D_2 - D_1)^2}{2S\sigma^2} \tag{2.9}$$

where λ is the shape parameter of the distribution and larger λ indicates less skewness. According to this model, the number of cycles until collapse depends on both systematic delay mismatch

and random jitter.

It should be noted that the model above does not consider supply noise, which could be very difficult to precisely model as described in [20]. Here, we consider only low frequency supply noise from power source or other circuits on chip. Following analysis in [20], supply noise adds correlated delay variations to all inverters in RO and can be viewed as an additional correlated variation to the $\Delta Delay_{i,A}$ and $\Delta Delay_{i,B}$ terms in equation 2.1 and 2.2. However, since the supply variation is common for all stages, variation of the delay difference between the 2 edges is not significant, which results in small fluctuations in the mean value of collapse time in equation 2.8. Despite the modulating of average cycles to collapse by supply noise, the cycles to collapse of a given run still follows inverse Gaussian distribution caused by thermal noise.

2.2.2 Systematic Mismatch vs. Random Jitter

As indicated by equation 2.8 and 2.9, the distribution of collapse time depends on the relative magnitude of systematic mismatch and random jitter. If systematic mismatch is small, noise will have a more significant impact, resulting in a longer collapse time with wider distribution. In this case, random bits can be obtained from the RO by recording the number of cycles to collapse. On the other hand, under large systematic mismatch, the RO will collapse in a few cycles with negligible variation. Such systematic behavior is unique for each die and therefore can be used to produce a chip ID or PUF [22] but is not useful for extracting random bits.

Typically systematic mismatch dominates in an even-stage RO and makes entropy extraction difficult. Hence, RO-based TRNGs have previously employed an odd-stage number RO where mismatch naturally cancels out [14]. However, we will show in Section 2.3 that, in fact, the process variation in an even-stage RO can be used as a natural source of tunability to enable a highly adaptive TRNG design that is robust to a wide range of environmental and other factors using an automatic tuning loop.

The relationship between process variation and tunability can be explained by our proposed model as well.$\Delta Delay_{i,A}$ and $\Delta Delay_{i,B}$ are the delay differences of each stage due to process variations, which follow independent normal distribution $(N(0,\sigma^2_{variation}))$. Therefore the denominator term $\sum_{i=1}^{S}(\Delta Delay_{i,B} - \Delta Delay_{i,A})$ in equation 2.8 also follows normal distribution

$(N(0, S \times \sigma^2_{variation}))$. The process variation results in a wide distribution of the mean value in equation 2.8, which forms the foundation for our tuning method based on device mismatch. Distribution of the inverse of a normally distributed random variable $(y = \frac{1}{x}, x\ N(0, \sigma_0^2))$ is,

$$pdf(y) = \frac{e^{-(\frac{2\sigma_0^2}{y^2})}}{\sqrt{2\pi}\sigma_0 \cdot y^2} \tag{2.10}$$

Combining equations 2.8 and 2.10, distribution of the average collapse time in equation 2.8 across process variations can be calculated as,

$$pdf(x_{mean}) = \frac{e^{-(\frac{2|D_2 - D_1|S\sigma^2_{variation}}{x^2_{mean}})}}{\sqrt{2\pi S}\sigma_{variation} \cdot x^2_{mean}}, x_{mean} > 0 \tag{2.11}$$

This distribution is highly skewed with long tails and matches the measured distributions in Section 2.4.2. The peak occurs at,

$$x_{peak} = \frac{|D_2 - D_1|}{\sqrt{2\pi S}\sigma_{variation}} \approx \frac{S \cdot Ideal\_Delay}{2 \cdot \sigma_{variation} \cdot \sqrt{2\pi S}} = \frac{S}{2 \cdot Ratio \cdot \sqrt{2\pi S}} \propto \sqrt{S} \tag{2.12}$$

where Ratio is the spread (σ/μ) of a single delay stage in the oscillator due to local process variations. As discussed previously, we need small systematic variation for random number generation which corresponds to larger average collapse time. A design implication from equationl 2.12 is that having more stages in RO increases our chance to get a RO with small enough systematic variation for TRNG.

2.2.3 Extracting Random Bits from Collapse Time

The analytical model above characterizes the behavior of oscillation collapse in even-stage RO and indicates that we can get larger variations in collapse time with less systematic variation in RO. To use the frequency collapse concept as entropy source for TRNG, the last step is extracting uniformly distributed random bits from collapse time following inverse Gaussian distribution. Our simple but efficient method shown in Figure 2.3 is to take the LSBs of the collapse count as outputs, which is based on dividing the distribution into small enough bins so that neighboring bins have negligible differences in probability. Similar strategies have already been applied in previous

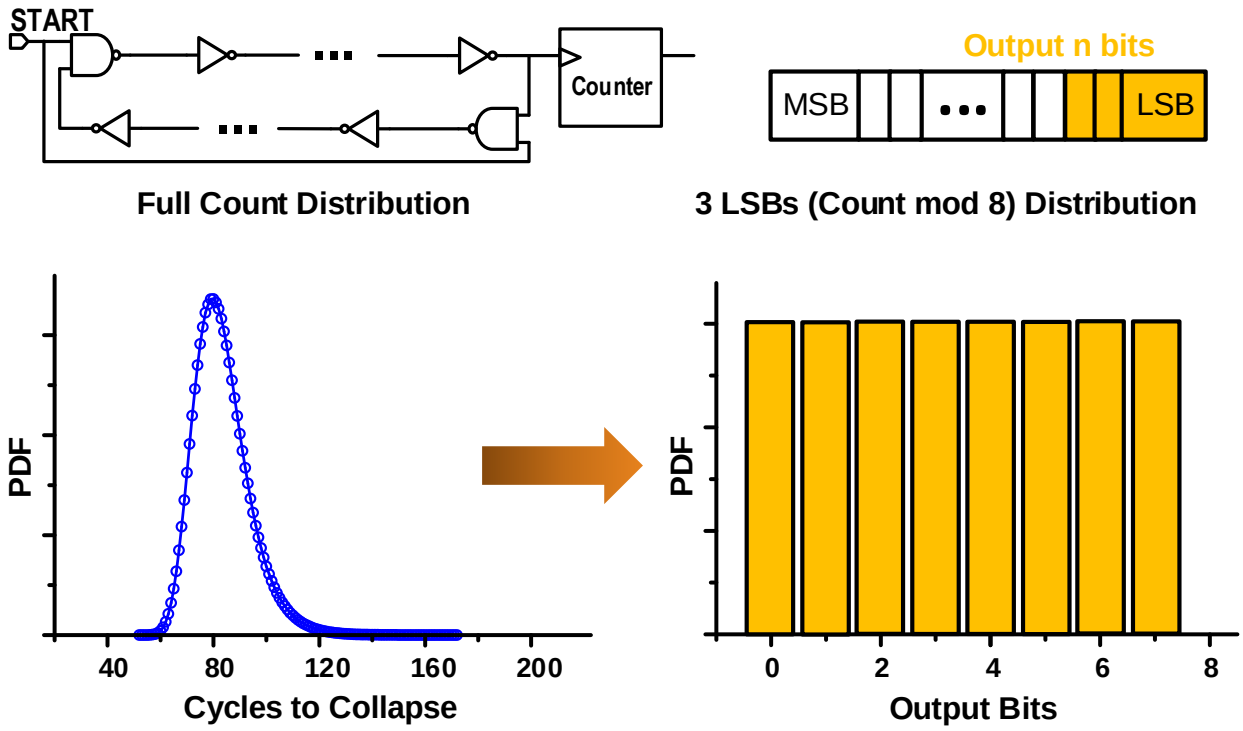


Figure 2.3: Random bit generation from collapse time.

TRNG designs [14, 15, 17]. The number of LSBs that can be used as random bits depends on the variance of the collapse time distribution, as shown in the measurement results in Figure 2.4. Small collapse time with small variations does not provide enough entropy and operation margins; while large collapse time results in low overall random bit throughput (defined as output frequency times number of useful bits) because the number of high entropy bits does not increase as fast as collapse time. Figure 2.4(b) shows number of random bits divided by average count as an approximation of throughput to show the desired range. Therefore we target a middle range of collapse time as a result of trade-offs.

2.3 All-digital Implementation of TRNG

The all-digital TRNG comprises only 3 parts, even-stage RO, control logic with a counter, and automatic tuning loop to counter process, voltage and temperature (PVT) variations. Such a simple design minimizes design efforts and offers good technology portability.

To make the RO working in the desired collapse condition discussed in the previous section,

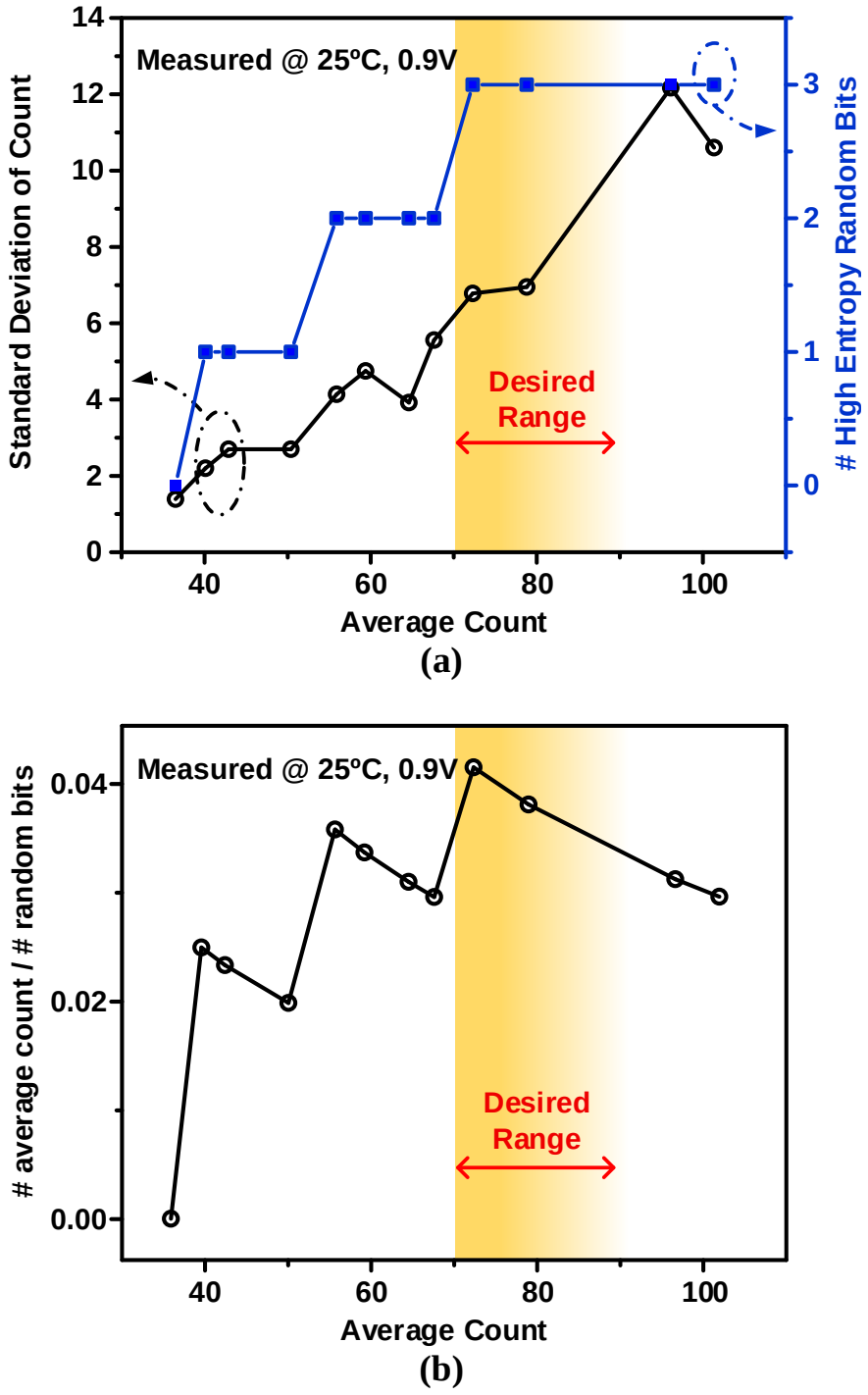


Figure 2.4: (a) Standard deviation of cycles to collapse and the number of high entropy random LSBs versus average collapse cycles. (b) Number of random bits divided by average count as an approximation of throughput to illustrate the desired range.

the proposed approach in Figure 2.5 replaces each inverter stage in the ring with a set of identically designed inverters and a multiplexer to select one of the inverters for the RO path as specified by configuration bits. Due to process variations, each inverter has slightly different delays in fabricated chips. Through different combinations of inverters, the delay differences between the two edges can be adjusted to meet collapse time requirement. Such a mismatch based tuning method introduces minimum overhead and provides fine enough tunability to satisfy the requirement. During startup, a simple control program described in Figure 2.7 running on the host processor reconfigures the RO as follows: An LFSR generates a random configuration trial and collects 500-5000 collapse times to calculate their mean and max values. If the mean collapse time is too low, systematic variations are not properly canceled out and a new configuration trial is attempted. Max count value is used to tune the system clock frequency so that most runs collapse within a given period. Mean and max collapse times that are out of range can also indicate intentional external attack as shown in the measurements in next section. Once the mean collapse time is in the correct range the RO is properly tuned and random numbers are generated while the host processor continues to monitor collapse times to adapt to any environmental changes.

As shown in Figure 2.5, the TRNG is implemented using a 32-stage RO with 8 selectable inverters per stage, providing a total of $8^{32} \approx 7.8 \times 10^{28}$ possible RO configurations. The selection of this configuration for prototype is a trade-off between 3 major considerations: (1) Enough tuning space is desired for robust design; (2) Shorter rings takes less area and has higher throughput because the RO runs faster; (3) According to the model in equation 2.12, peak of the average collapse time distribution increases with more stages in the ring, which increases the possibility to find configurations with larger collapse count.

The RO is reset during negative phase of the clock (CLK) and started by the rising edge of CLK. A 9b counter records the cycles to collapse (COUNT), which is sampled at the rising edge of CLK. Edge collapse automatically stops the counter, eliminating the need for extra phase/frequency detectors (PFD) and other peripheral circuits as in [14] and [15] , saving power, area, and potential non-idealities caused by PFD. The frequency of CLK should be chosen to allow most runs collapse in one CLK period. If the RO doesnt collapse, COUNT will be a fixed maximum value and causes bias in generated random bits. To eliminate this negative impacts, programmable SR latches are included in counter, which sets an invalid flag once the counter value reaches the

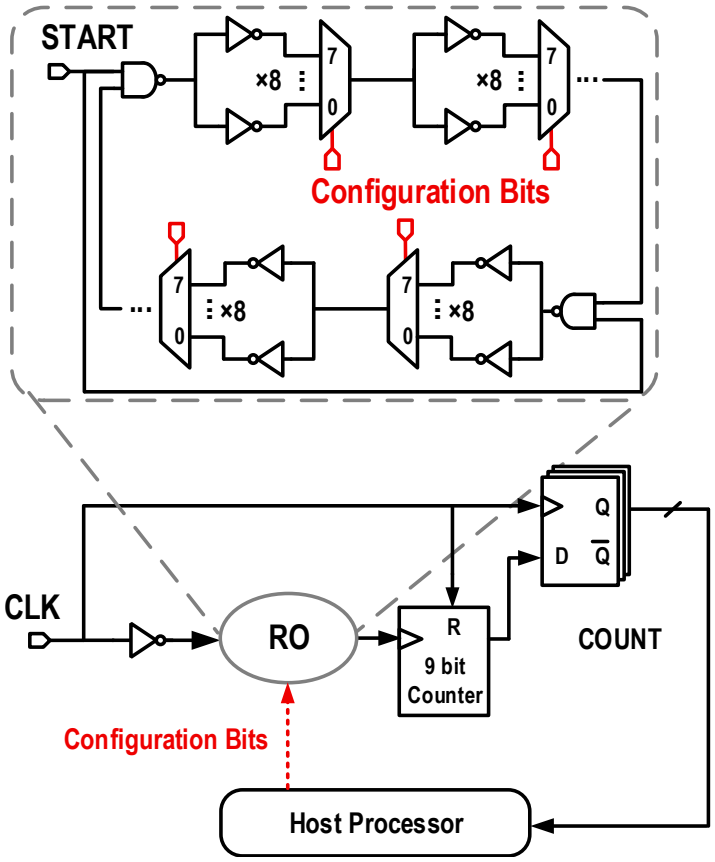


Figure 2.5: TRNG block diagram and tunable RO with eight inverters per stage.

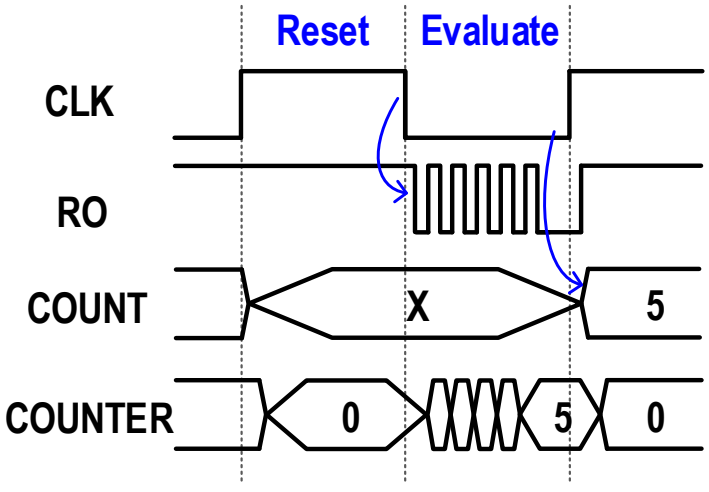


Figure 2.6: Operating waveform of TRNG.

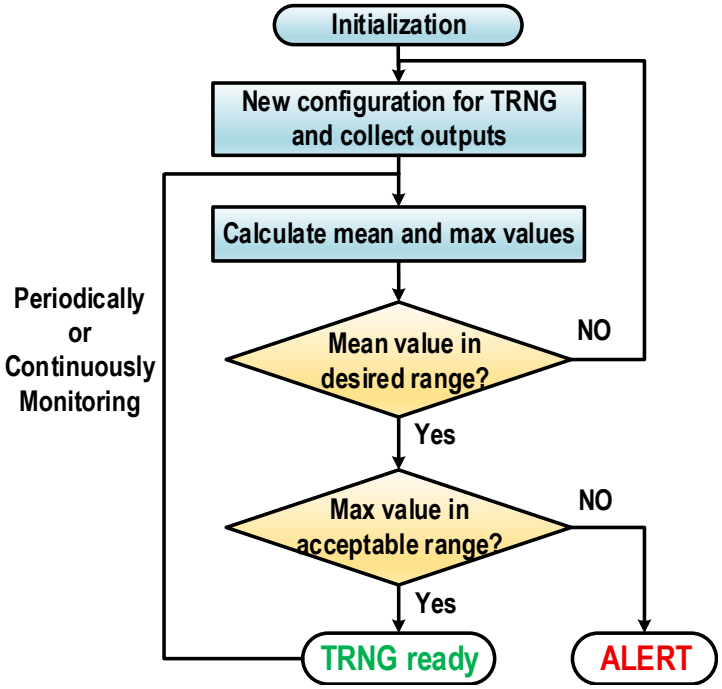


Figure 2.7: Automatic tuning FSM of TRNG.

programmed value. Since TRNGs are typically co-located with a SoC processor, the tuning algorithm can run on the host-processor (off-chip in our tests) although its simplicity makes it suitable for hardware implementation.

2.4 Measurement Results

The all-digital TRNG is implemented in 40nm CMOS GP technology with a nominal voltage of 0.9V. Measurement results in this section except section 2.4.5 are all based on the 40nm prototype. For many mobile and IoT applications, however, older technologies are used because of lower power and cost. To show the portability of the design and the functionality of the TRNG in an older technology with less process variation and noise, a second prototype is fabricated in 180nm CMOS technology. Measurement results of the 180nm chip are provided in Section 2.4.5. Figure 2.8 shows the die micrographs of the 40nm chip with a core area of 836 μm^2 and 180nm chip with a core area of 7250 μm^2.

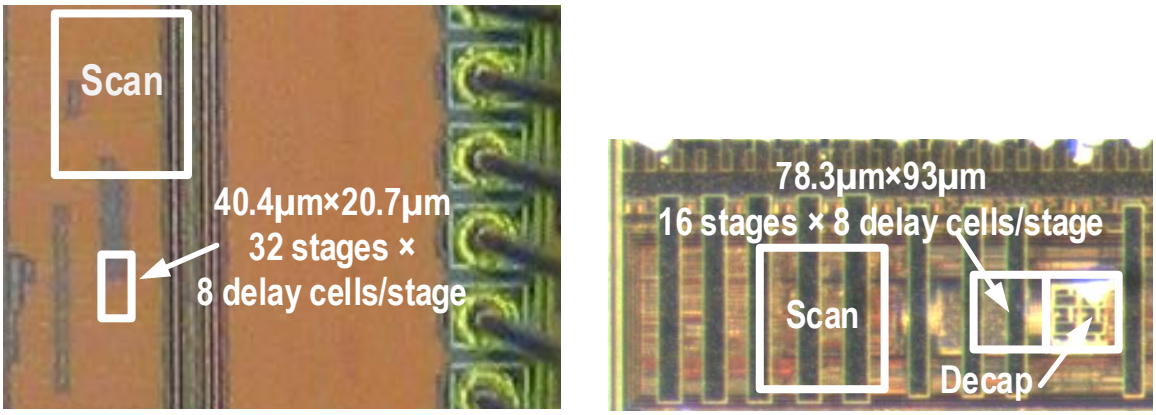


Figure 2.8: Die micrographs of 40 and 180 nm TRNG prototypes.

2.4.1 Randomness and Performance of the TRNG

The randomness of the test chip is evaluated by NIST Pub 800-22 RNG testing suite (15 tests) with recommended settings and 100Mb raw data for each run [5]. The TRNG is robust and passes all NIST tests across all combinations of voltage (0.6 to 0.9V in 100mV steps) and temperature (-40 to 120$^{\circ}C$ in 30$^{\circ}C$ steps) with a required mean-count range of 70 to 90 cycles. As shown in Figure 2.9, at lower temperature, the spread (σ/μ) of collapse count is lower due to less thermal noise, but the target mean count range ensures sufficient quality of 3 LSBs even at -40$^{\circ}C$, while enabling successful RO tuning within an acceptable number of configuration trials. Table 2.1 shows NIST test suite results of 5 chips at one of the worst-case conditions (0.6V, -40$^{\circ}C$). Throughput is decided by number of extracted bits per run and system clock frequency. The two factors are contradicting to each other. Number of high quality bits is decided by the target collapse condition, larger collapse counts provide more random bits but require slower system clock. Measurement shows that finding a region to extract 3 bits is optimal. Once the target region and number of extracted bits are decided, required system clock frequency and overall throughput can be determined based on RO frequency. The dependence of throughput on environmental conditions is same as that of a single RO. Figure 2.10 shows throughput of the TRNG ranges from 300kbps to 2Mbps and energy efficiency from 8.7 to 37.2 pJ/b across 0.5 to 1V at 25$^{\circ}C$. A summary of measurement results and comparisons to prior works are given in Table 2.3.

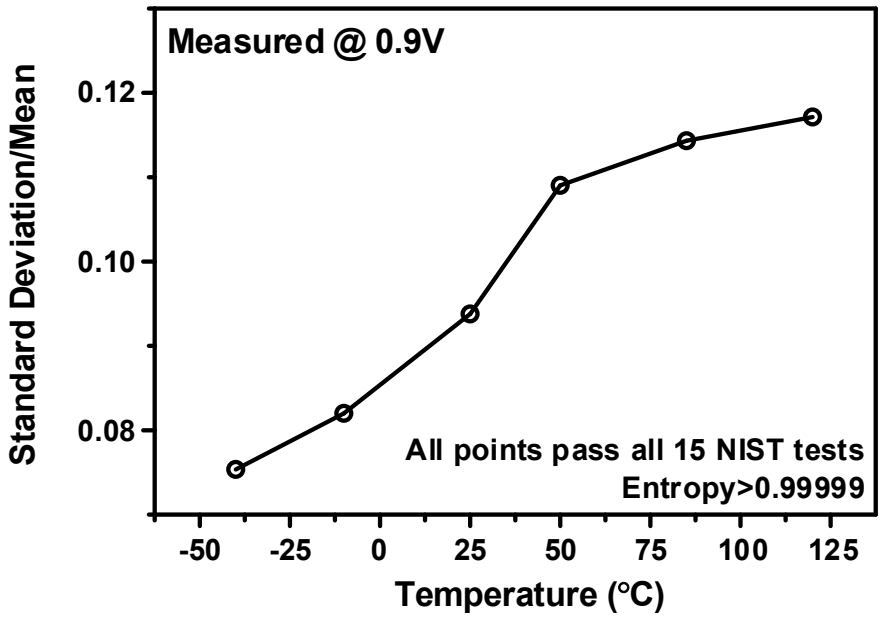


Figure 2.9: Measured spread (standard deviation/mean) of cycles to collapse across temperatures.

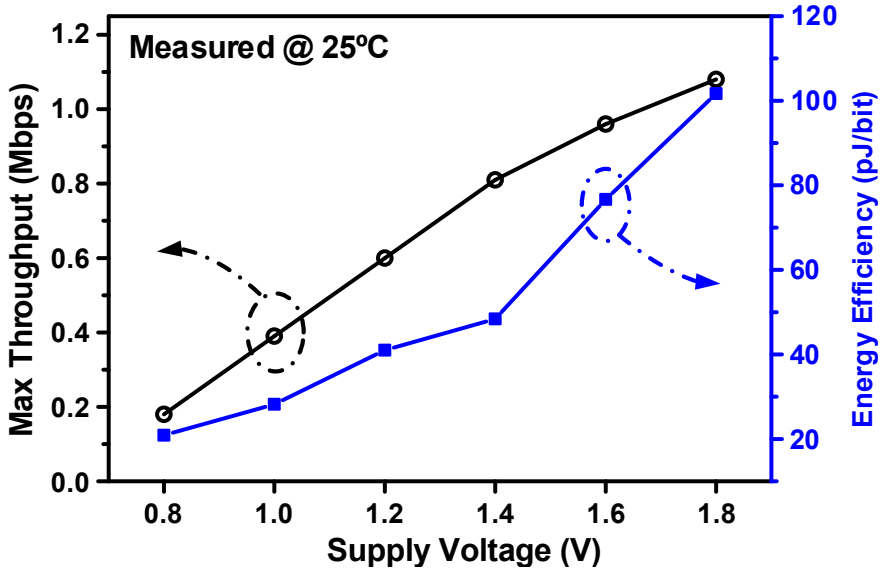


Figure 2.10: Measured impacts of supply voltage on throughput of TRNG.

Table 2.1: Measured NIST test suite results of five chips at worst case condition (-40°C, 0.6V)

NIST Pub 800-22, Randomness Test	Chip #1		Chip #2		Chip #3		Chip #4		Chip #5	
	21 trials		102 trials		34 trials		217 trials		63 trials	
	Pvalue	Pass	Pvalue	Pass	Pvalue	Pass	Pvalue	Pass	Pvalue	Pass
Frequency	0.69	0.987	0.28	0.990	0.13	0.993	0.97	0.993	0.13	0.993
Block Frequency	0.12	0.983	0.03	0.983	0.43	0.993	0.45	0.987	0.20	0.987
Cumulativ Sum (1)	0.57	0.983	0.09	0.990	0.55	0.987	0.02	0.993	0.25	0.993
Cumulativ Sum (2)	0.69	0.993	0.69	0.990	0.28	0.990	0.36	0.990	0.60	0.990
Runs	0.16	0.990	0.90	0.987	0.63	0.997	0.40	0.977	0.28	0.983
Longest Runs	0.70	0.990	0.98	0.997	0.86	0.990	0.44	0.993	0.93	0.990
Matrix Rank	0.02	0.993	0.12	0.990	0.22	0.983	0.11	0.987	0.12	0.990
FFT	0.39	0.983	0.24	0.980	0.20	0.990	0.52	0.993	0.44	0.993
Serial (1)	0.88	0.983	0.72	0.987	0.84	0.993	0.03	0.990	0.84	0.993
Serial (2)	0.93	0.983	0.21	0.990	0.29	0.987	0.05	0.983	0.57	0.983
Linear Complexity	0.93	0.987	0.17	0.990	0.60	0.990	0.40	0.983	0.03	0.977
Non Overlapping Template	PASS	PASS	PASS	PASS	PASS	PASS	PASS	PASS	PASS	PASS
Overlapping Template	0.19	1.000	0.63	0.980	0.65	0.980	0.57	0.970	0.83	0.980
Universal	0.21	0.990	0.38	0.970	0.26	0.980	0.70	0.960	0.49	0.990
Random Excursions	PASS	PASS	PASS	PASS	PASS	PASS	PASS	PASS	PASS	PASS
Random Excursions Variant	PASS	PASS	PASS	PASS	PASS	PASS	PASS	PASS	PASS	PASS
Approximate Entropy	0.03	0.960	0.35	0.970	0.26	0.980	0.43	0.990	0.55	1.000

* "PASS" means all sub tests pass minimum requirement.

** Minimum p-value χ^2 is 0.0001. Minimum pass rate is 0.97 for first 10 tests (using 300 × 40K bits) and 96/100 for the other 5 tests (using 100 × 1M bits).

+"PASS" means all sub tests pass minimum requirement.

++ Minimum p-value χ^2 is 0.0001. Minimum pass rate is 0.97 for first 10 tests (using $300 \times 40K$ bits) and $96/100$ for the other 5 tests (using $100 \times 1M$ bits).

Table 2.2: Hit rate of random search under different environmental conditions

Temp	VDD	Possibility in 70-90 range
25	0.9	8%
-40	0.9	6%
-40	0.6	5%
120	0.9	5%
120	0.6	6%

Table 2.3: TRNG performance summary and comparison with recent publications

	This work				CICC 14 [15]	ISSCC 14 [14]	JSSC 12 [10]	VLSI 11 [17]	JSSC 08 [11]	ISSCC 06 [8]
	0.9V	0.6V	1.8V	0.8V						
Technology	40nm		180nm		65nm	28nm	45nm	65nm	130nm	120nm
Entropy Source	Jitter in oscillator				Jitter in oscillator	Jitter in oscillator	Meta-stability	Time to oxide breakdown	Meta-stability	Meta-stability
Bit Rate (Mb/s)	2	0.45	1.08	0.18	2	23.16	2400	0.011	0.2	0.2
Tested Operating Conditions	0.6 to 1V -40 to 120 °C		0.8 to 1.8V		0.8 to 1.2V	N/A	0.28 to 1.35V	N/A	N/A	N/A
NIST Pass	All				All	All	All	All	5	-
Area (μm^2)	836		7250		6000	375	4004	1200	36300	9000
Power (mW)	0.046	0.005	0.109	0.0037	0.13	0.54	7	2	1	0.05
Efficiency (nJ/bit)	0.023	0.011	0.101	0.021	0.066	0.023	0.0029	181.81	5	0.25
Post Processing	No				No	No	No	No	No	Yes
Frequency Attack Robustness	Yes (up to 500mV)				N/A	Yes (filter)	N/A	N/A	N/A	N/A

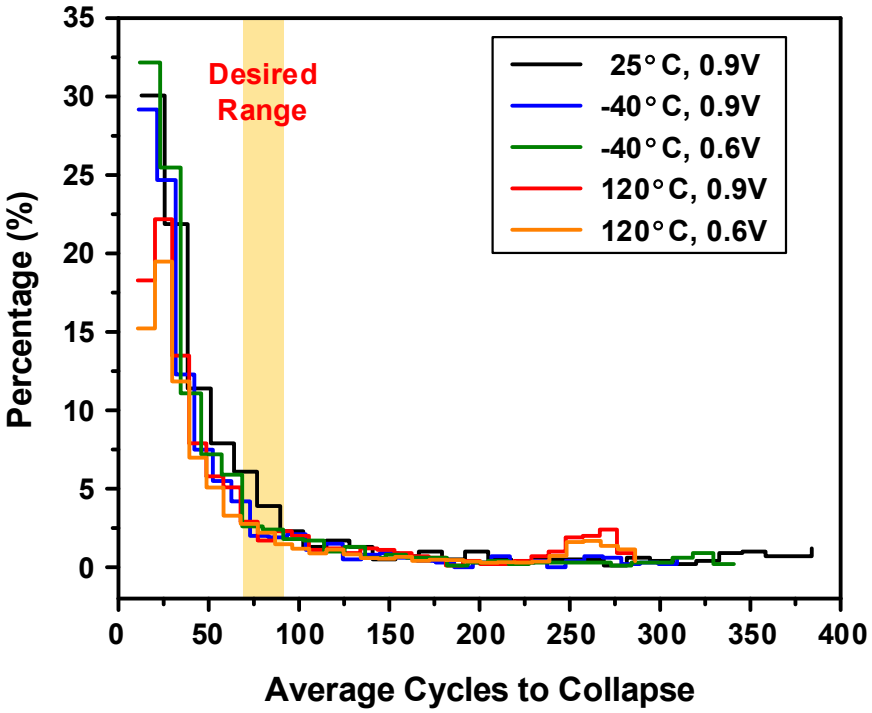


Figure 2.11: Measured distributions of average cycles to collapse across random configurations.

2.4.2 Random Search Performance of the Tuning Loop

Since the tuning loop is based on random search, the number of trials to achieve desired configuration is random and affects the setup time of the TRNG. For each RO configuration, an average collapse time can be measured. Distributions of this value across 5000 configurations and 5 environmental conditions are shown in Figure 2.11. It can be seen that distributions are very similar across environmental variations. Small differences in distributions come from different variance of inverter delay under different conditions. Inverters have larger delay variations at lower VDD and lower temperature, according to simulation results. From the measured distribution, the possibility a random configuration falls in desired range is calculated and shown in Table 2.2. Assume the trials are random and independent, the probability that first success occurs at nth trial is,

$$P(n) = (1-p)^{n-1}p \tag{2.13}$$

where p is the probability of success for one trial. Here, the worst case can be approximated as 5% from Table 2.2. The expectation of the distribution in equation 2.13 could be calculated as $1/p$ and therefore it should take 20 trials on average to find a proper configuration hitting target range.

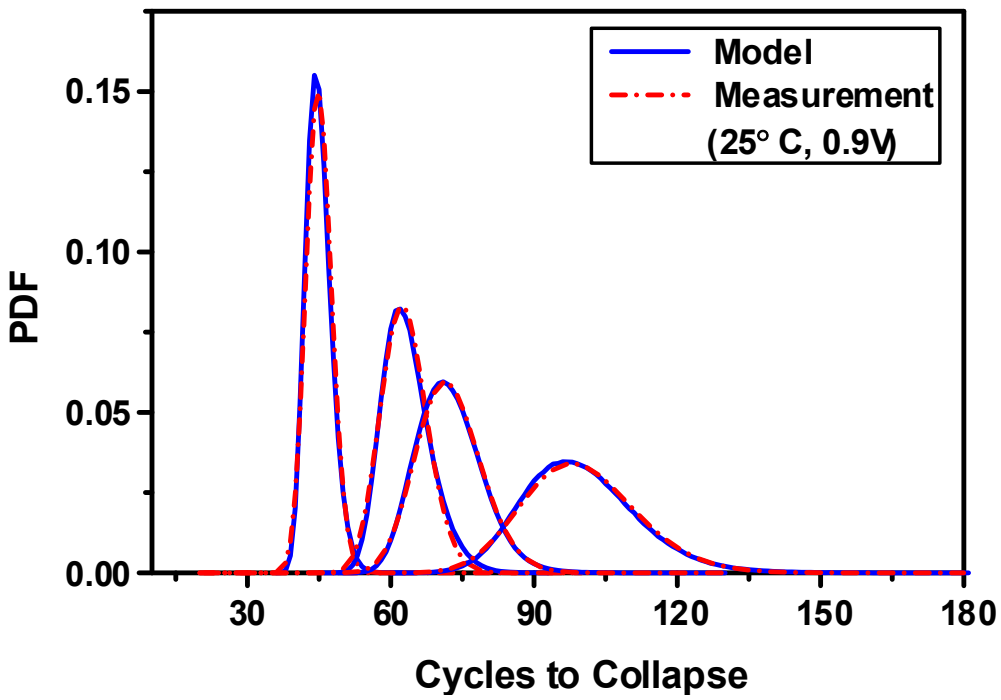


Figure 2.12: Measured distribution of cycles to collapse versus analytical model derived from measured mean and variances.

In addition to this statistical approximation, it was found that the worst case throughout the whole testing process was 315 trials to meet targets.

2.4.3 Validation of TRNG Analytical Model

Measurement results also confirm the validation of the TRNG model presented in Section 2.2.1. In Figure 2.12, the measured distribution of cycles to collapse with different settings are shown in red lines while inverse Gaussian distribution calculated from measured mean and variance values are shown in blue lines. The correspondence of the two lines confirm the distribution of collapse time proposed in our model.

For simplicity and small area, no voltage regulation or dedicate de-coupling capacitors (decaps) are included on-chip and no external regulator is added to testing board, which emulates a worst-case noisy environment in SoC. In practical applications, decaps and possibly voltage regulators can be added to suppress supply noise. As discussed in Section 2.2, supply noise affects average collapse time causing fluctuations in the distribution. To better illustrate the effects, 3 random runs at nominal 0.9V and 25°C with different configurations are divided into 1000 partitions (1K runs in

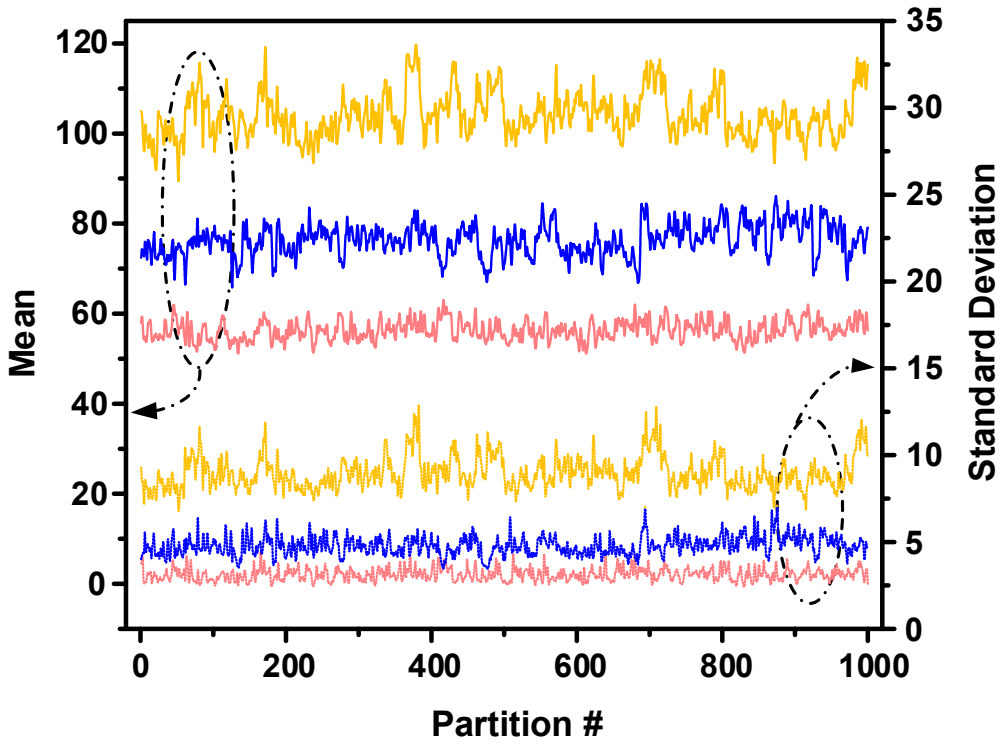


Figure 2.13: Fluctuation of collapse time mean and standard deviation values of three runs with different collapse times.

each partition) with average values and standard deviations of each partition plotted in Figure 2.13. Fluctuation of average collapse time exists in our measurement with a spread (σ/μ) around 5%. In fact, all testing results in Section 2.4 are measured in same noisy environments and proved the overall distribution can still be approximated by inverse Gaussian distribution and is capable to generate high quality random bits.

2.4.4 Supply Noise Injection Attack to the TRNG

Supply noise injection is an effective attack technique to compromise TRNGs by injection locking [18]. Supply noise at multiples of RO frequency locks the oscillation with smaller jitter and greatly reduces entropy harvested by conventional RO based TRNG. To test the robustness of the proposed TRNG to injection locking, we implement a noise injection testing setup by coupling a sine wave to the DC supply voltage (Figure 2.14).

As shown in Figure 2.15, injection locking occurs at harmonics of the ring frequency (f_{RO}) and impacts both the collapse count and bit entropy. Here, the injection locking not only reduces

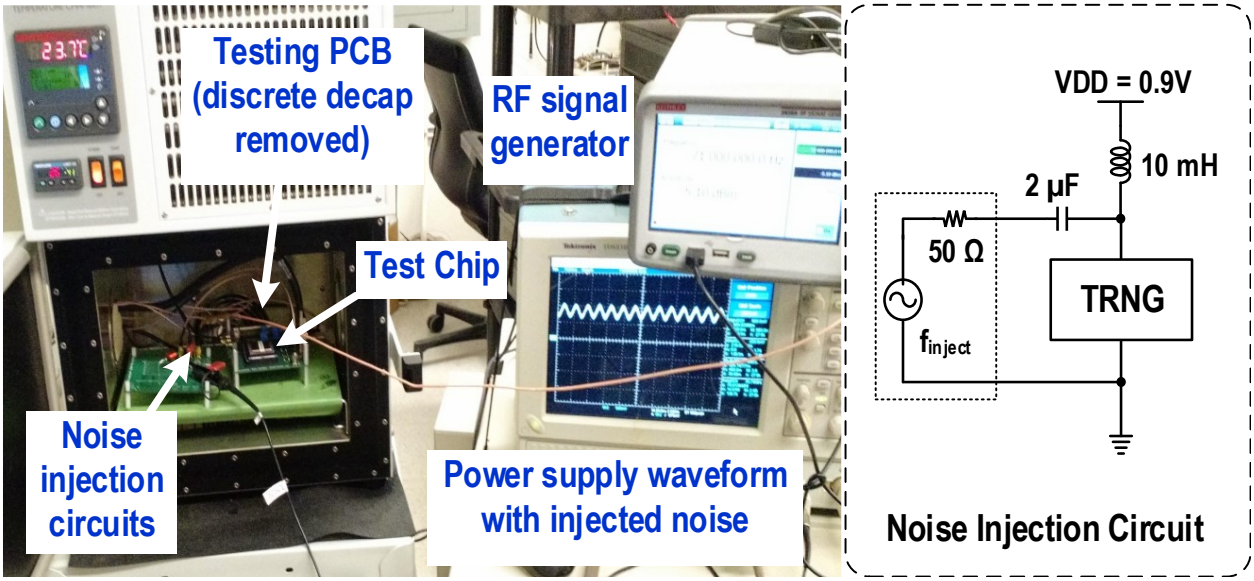


Figure 2.14: Supply injection attack testing setup and schematic of noise injection circuits.

the jitter of the oscillation, but also "locks" the distance between the 2 edges in the RO, causing the RO to collapse slower. When directly applying such a supply noise injection attack to a single configuration of the TRNG, it fails to pass NIST tests. Figure 2.15 and Figure 2.16 illustrate the impacts of injected supply noise frequency and amplitude on the average collapse time. Shannon entropy of generated random bits are shown by the right y-axis as a simple indicator of the randomness under different conditions. As can be seen, injection locking happens at multiples of RO frequency, and the effect is most severe at $3 \times f_{RO}$ and begins to degrade randomness after supply noise peak-to-peak amplitude is larger than 250mV. However, since the injection locking shifts the mean collapse count outside the specified range and changes the distribution of cycles to collapse (Figure 2.17), the control loop can automatically detect and reject the harvested bits. It then selects new configurations that provide slightly different oscillation frequencies, restoring the desired average count value and randomness. Hence, while operating the control loop, all NIST tests are passed with injection peak-to-peak amplitudes up to 500mV at the worst-case injection frequency of $3 \times f_{RO}$. If more sophisticated attacks are used, it is possible the tuning loop cannot restore normal operation of the TRNG but the attack can still be detected to minimize damage to the secure system.

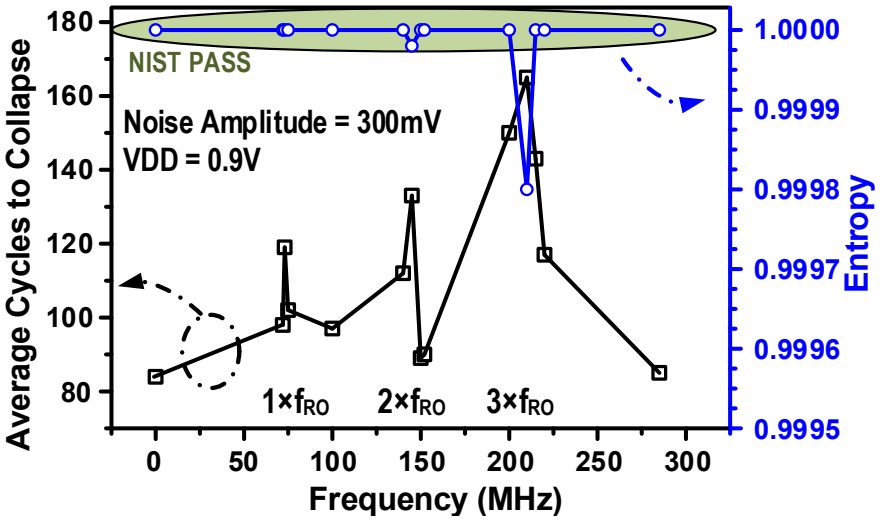


Figure 2.15: Measured impacts of supply noise frequency on randomness of the TRNG.

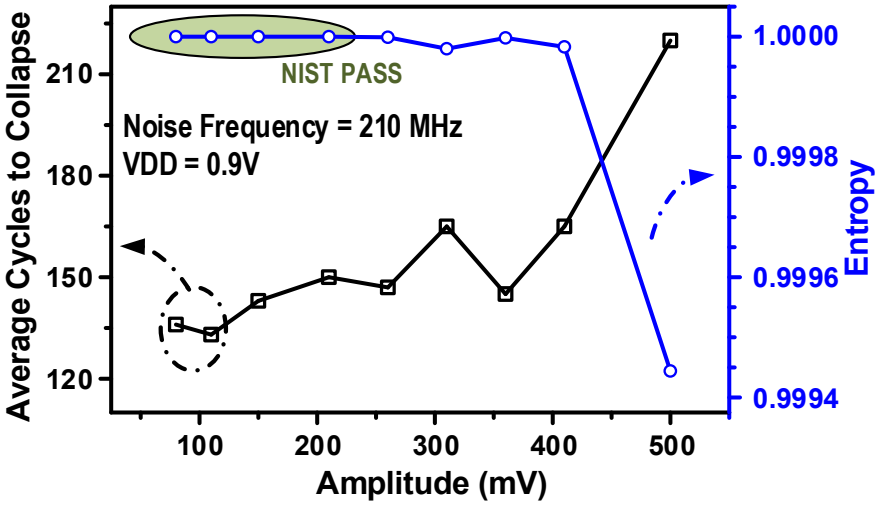


Figure 2.16: Measured impacts of supply noise amplitude on randomness of the TRNG.

2.4.5 Measurement Results of 180nm TRNG Prototype

180nm devices have much smaller conducting current and therefore have less delay variation due to noise, which poses difficulties to TRNG designs. Measurement results shows that a configuration with 75 average cycles to collapse has a standard deviation of 4 in 180nm, while in 40nm the standard deviation is 6.8. To overcome the decrease in variance, RO need to be configured to a operating region with 80-100 average cycles to collapse, which ensures the entropy of 3 LSBs. Figure 2.18 shows measured throughput from 1.08 Mbps to 0.18 Mbps and energy efficiencies from 101.7 pJ/b to 28.9 pJ/b across 1.8V to 0.8V supply voltages. NIST tests confirm the randomness

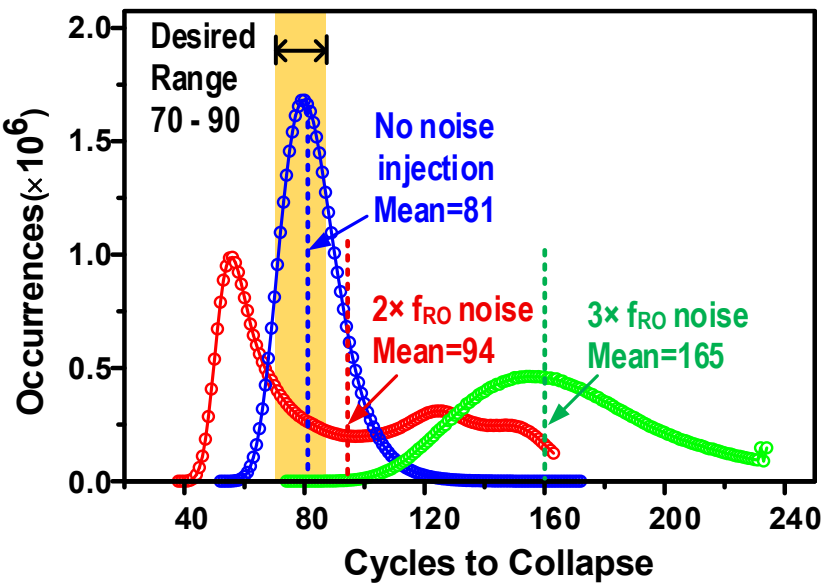


Figure 2.17: Cycles to collapse distribution before and after supply noise injection of 300 mV.

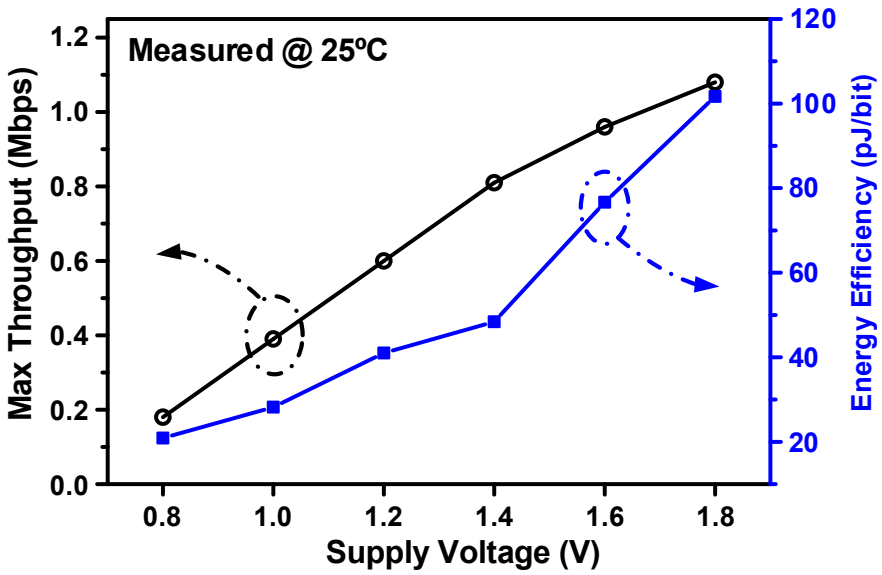


Figure 2.18: Measured throughputs and energy efficiencies across supply voltages for 180 nm chip.

of the design from 1.8V down to 0.8V. A summary of the 180nm test chip measurement results is included in Table 2.3.

Another concern about the TRNG in 180nm is less process variation, which is used to tune collapse condition in the proposed design. Figure 2.19 shows the distributions of average cycles to collapse at different supply voltages. It can be seen that lower supply voltage shift the distribution left and make it less possible to find a proper configuration. Compared to 40nm results in

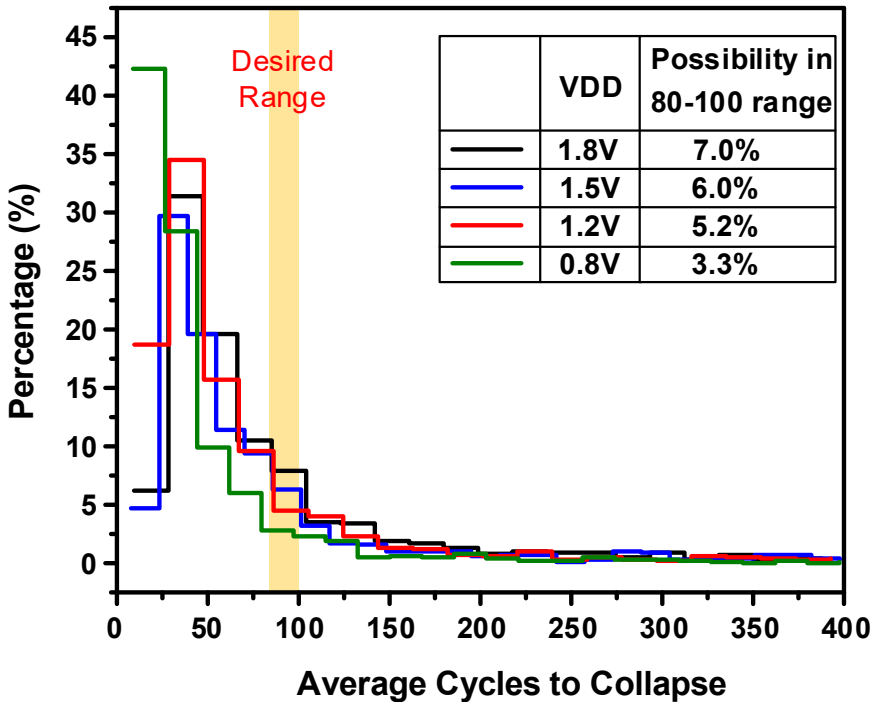


Figure 2.19: Measured distributions of average cycles to collapse across random configurations for 180 nm chip.

Figure 2.11, less process variation shifts the distribution right which compensates the increase in target values. The overall hit rate of random search is similar to that of 40nm design.

2.5 Summary

This work presents an all-digital true random number generator (TRNG) harvesting entropy from the frequency collapse event of two edges injected into an even-stage ring oscillator. The cycles to collapse serves as a good indicator of the quality of generated random bits. A configurable ring based on device mismatch and an automatic tuning loop based on cycles to collapse provides robustness across a wide range of temperature (-40 to 120°C), voltage (0.6 to 0.9V), process variation, and external attack. Due to the simplicity and robustness of the tuning scheme, no fine-grained tuning circuits or extensive voltage regulation is needed. Therefore, it is simple to port this all-digital design to other technologies. Measurement results prove that the TRNG works in 180nm CMOS technology commonly used for ultra-low-power sensor applications. Tested chips pass all NIST randomness tests across all measured operating conditions and power supply attacks.

CHAPTER 3

Fully Synthesized True Random Number Generator

3.1 Introduction

Chapter 2 introduces a robust silicon true random number generator (TRNG) for secret key generation, which requires a feedback loop to adjust the TRNG core to be robust against process, voltage and temperature variations. In this chapter, in order to further simplify the design of TRNGs for better technology portability and smaller footprint, the concept of frequency collapse in multi-mode ring oscillator introduced in Chapter 2 is extended to odd-stage ring oscillators. The new design eliminates the effects of process variations and can be implemented with only standard cells and automatic placement & routing tools. The designs fabricated in 28nm and 65nm can consistently pass all NIST randomness tests.

The remainder of the chapter is organized as follows. The concept of using frequency collapse in an odd-stage RO as entropy source for true random number generation is described in Section 3.2; detailed implementation of the TRNG prototype using only standard cells and commercial tools is shown in Section 3.3; measurement results of 40nm test chips are provided in Section 3.4; and finally, the paper is concluded in Section 3.5.

3.2 Frequency Collapse in Odd-Stage Ring Oscillator

In a conventional odd-stage ring oscillator (RO), the start signal injects one edge that propagates through the ring to generate oscillation, as shown in Figure 3.1. However, if multiple edges

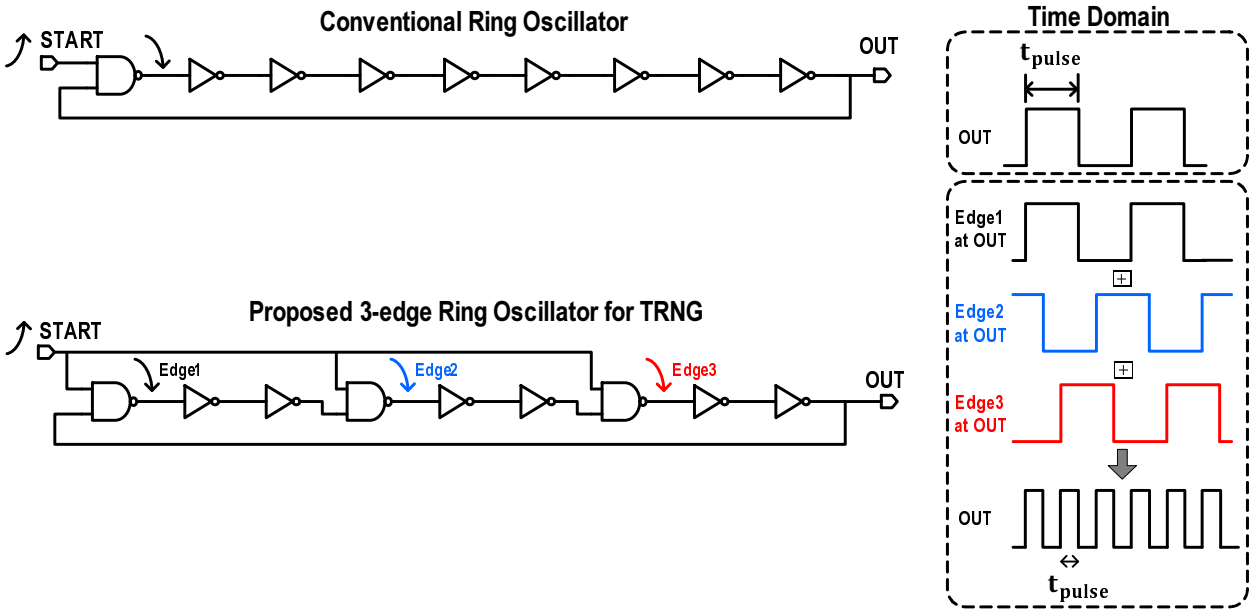


Figure 3.1: 3-edge ring oscillator running at 3rd order harmonic frequency.

are injected into a long enough ring, higher oscillation frequencies can also be supported. For example, three edges can be injected into a ring simultaneously to generate a 3 times faster oscillation frequency (Figure 3.1). Each edge propagates as in a conventional RO; the period of each edge is the same, but the three edges are 120° phase shifted and overall frequency is boosted by $3/times$.

The pulse width of generated waveform depends on systematic delay of the ring as well as jitter in the oscillator. For a conventional RO, the pulse width is only affected by the most recent cycle, so that the variance of pulse width will not increase over time. Comparatively, as shown in Figure 3.2, the three edges independently accumulate jitter from thermal noise, causing an increasing variation of the pulse width with each completed cycle.

Given time, two of the three edges will eventually meet and collapse due to opposite phases of neighboring edges, forcing the RO back to its nominal $1\times$ frequency mode (Figure 3.3). The time it takes to collapse reflects the accumulation of jitter and is used as the entropy source for random number generation. In contrary to the even-stage RO introduced in Chapter 2, all three edges go through the same path so systematic delay variation is inherently canceled. This property enables a TRNG design that can tolerate delay cell and interconnect variations. Therefore, the TRNG can be synthesized with only standard cells and implemented with standard placement & routing tools.

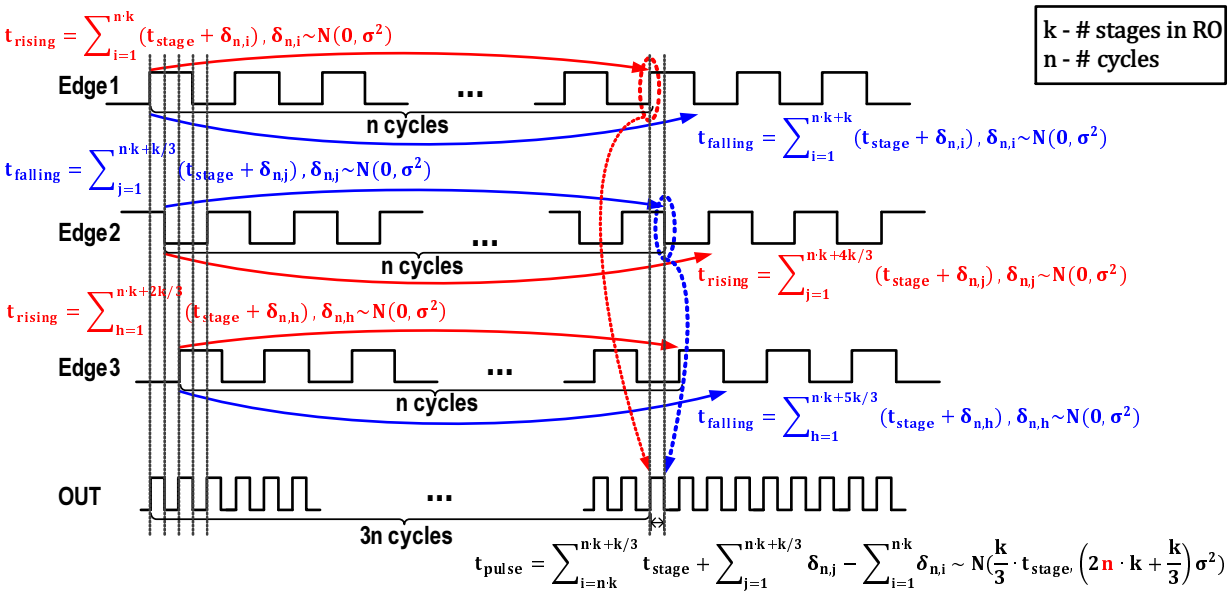


Figure 3.2: Pulse width of 3-edge ring oscillator.

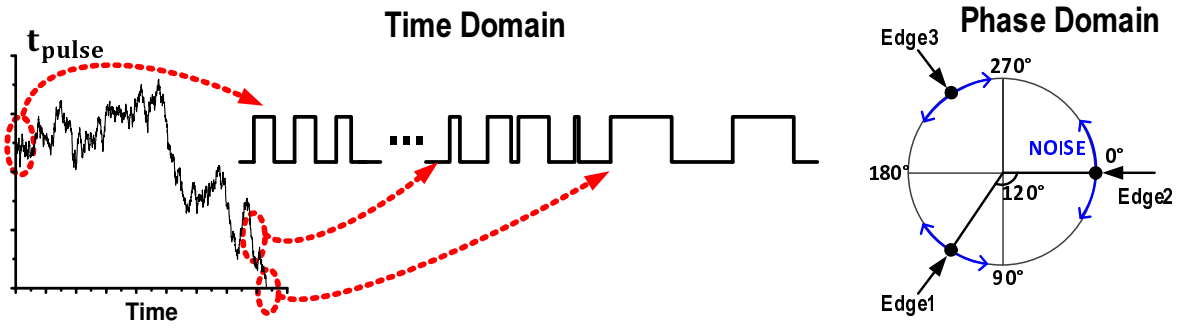


Figure 3.3: Frequency collapse of the 3-edge ring oscillator in time and phase domains.

3.3 Implementation of Fully Synthesized TRNG

Figure 3.4 shows the TRNG implementation consisting of 2 ROs, a counter, and control logic. A conventional RO (RO_REF) with 2/3 as many stages as the 3-edge RO (RO_RNG) acts as a reference for the phase frequency detector (PFD) to determine the edge collapse event. Since the frequency change is large (3×), a conventional digital implementation of the PFD is used, which enables a fully synthesizable design. To avoid setup and hold time violations in the sampling registers, a glitch removal stage and 2-bit shift register is added. This ensures that a collapse event is flagged only after two consecutive pulses. A 14-bit cycle counter triggered by the 3-edge RO records the number of cycles till collapse. An intermediate counter bit, COUNT [8], is used to

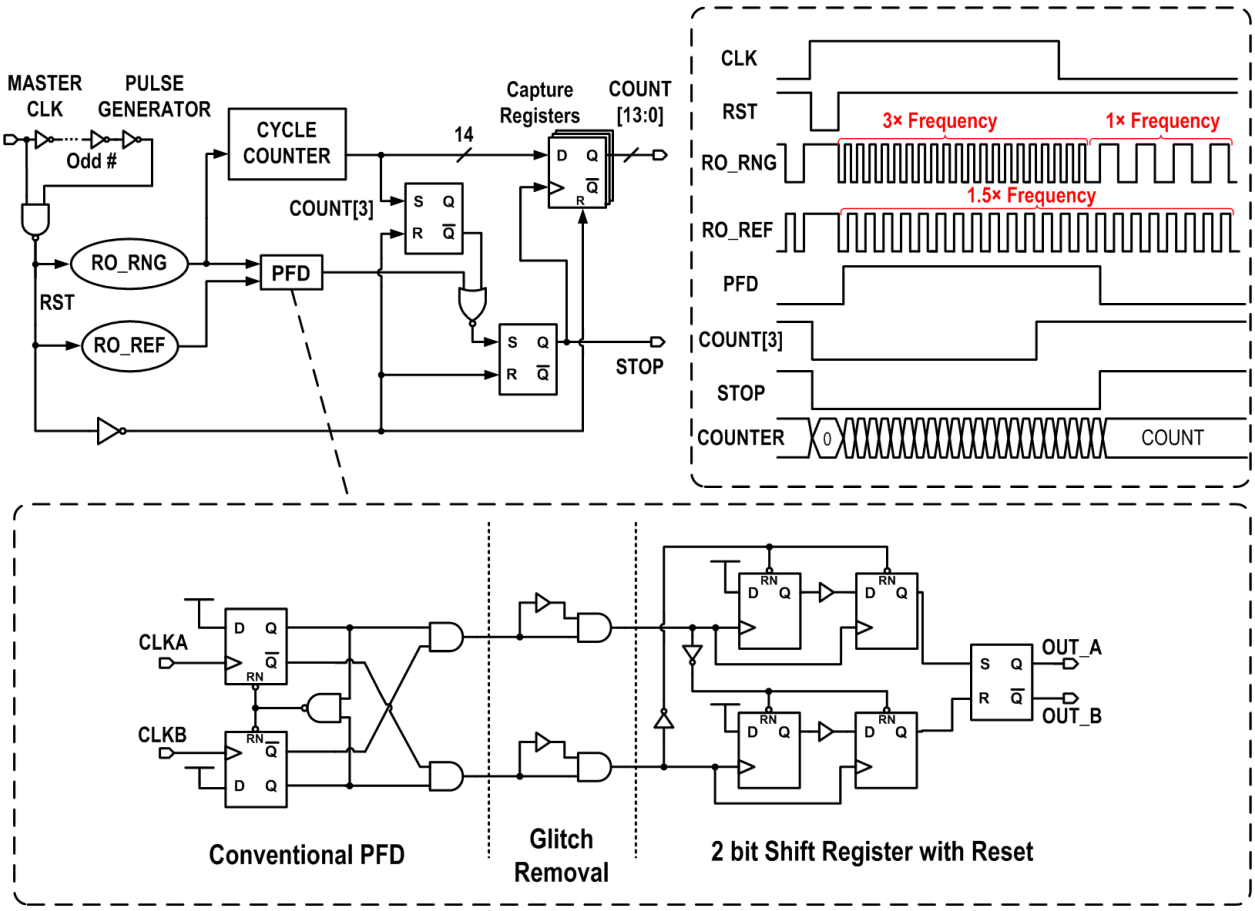


Figure 3.4: TRNG system block diagram and phase frequency detector (PFD) implementation

prevent false triggers in the first few cycles. Random number generation is initiated by a master clock, which is set sufficiently slow to ensure that the vast majority of collapse events (e.g., $> 90\%$ in the tested design) complete within the active phase duration. TRNG throughput is determined by the master clock frequency and the number of random bits harvested from each collapse event. The capture register reads the cycle counter when triggered by the PFD. As expected, the collapse cycle count follows log-normal distribution (Figure 3.5). To transform this into a uniform distribution, the collapse cycle counter is truncated, retaining the lower p bits while the LSB is dropped to eliminate sensitivity to mismatch in the counter sampling flip-flop.

All hardware TRNGs must consider interference from a potentially noisy environment as well as dedicated attacks. ROs are known to be sensitive to frequency injection, which can introduce errors in RO-based TRNGs [18]. The proposed TRNG uses accumulated jitter rather than jitter at

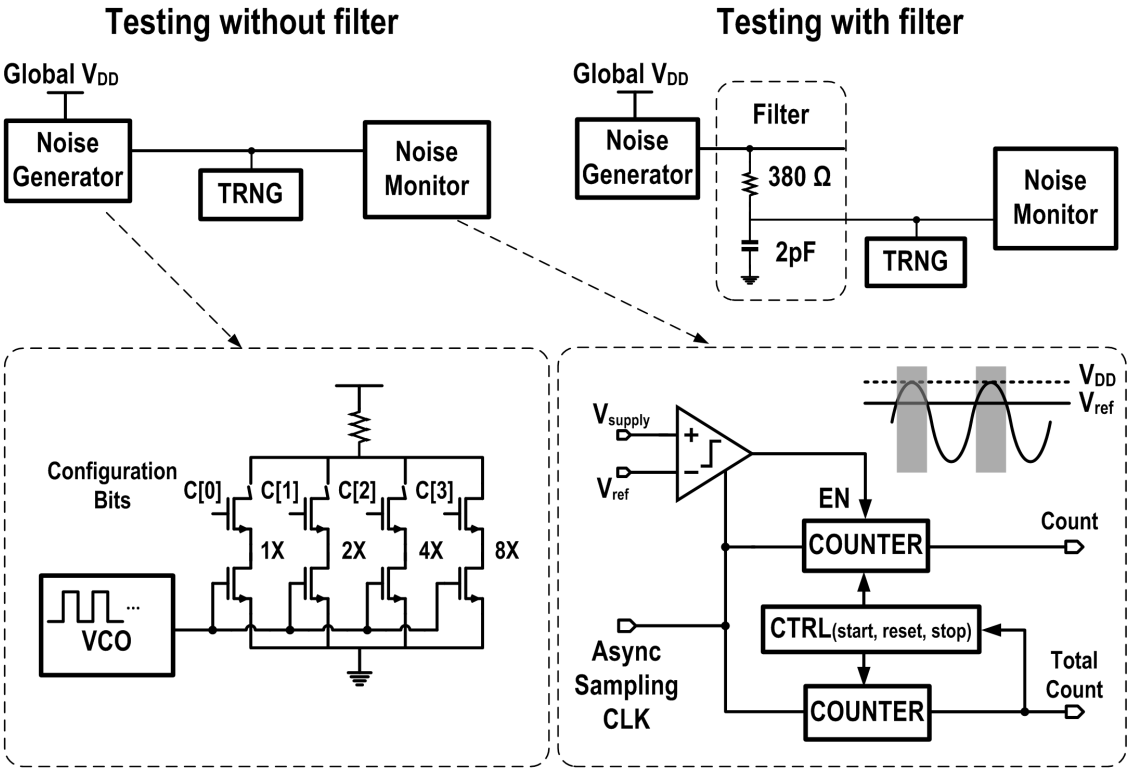


Figure 3.5: On-chip supply noise testing setups for protected and unprotected TRNGs.

a specific time point, making it more robust to noise injection. We tested the TRNG's sensitivity to deliberate attack with off-chip noise sources and also created on-chip test structures to inject and measure noise (Figure 3.5). A programmable noise generator controlled by an on-chip VCO introduces substantial noise to the TRNGs supply, locking the oscillation and impacting collapse event time. To measure noise amplitude on-chip, an asynchronous clock samples the supply voltage, compares it with an external reference voltage, and increments a counter accordingly. With sufficient samples (2^{14} here) the noise amplitude can be determined from the counter value. In addition, an RC filter with 210MHz corner frequency is designed to mitigate the impact of supply noise (Figure 3.5).

3.4 Measurement Results

The TRNG is evaluated with two test chips as shown in Figure 3.6; one in 28nm CMOS with 8 different rings; the other in 65nm CMOS with 48 different TRNGs. The NIST Pub 800-22 RNG testing suite is used to evaluate the randomness of generated bits with 112M bits in total across 15 tests. Both 28nm and 65nm TRNGs pass all 15 NIST tests as shown in Figure 3.7. Shorter rings with higher frequency collapse faster but have a narrower distribution, reducing the number of random bits obtained per cycle (i.e., require higher truncation). Longer rings provide more random bits but overall throughput is limited by the slower master clock.

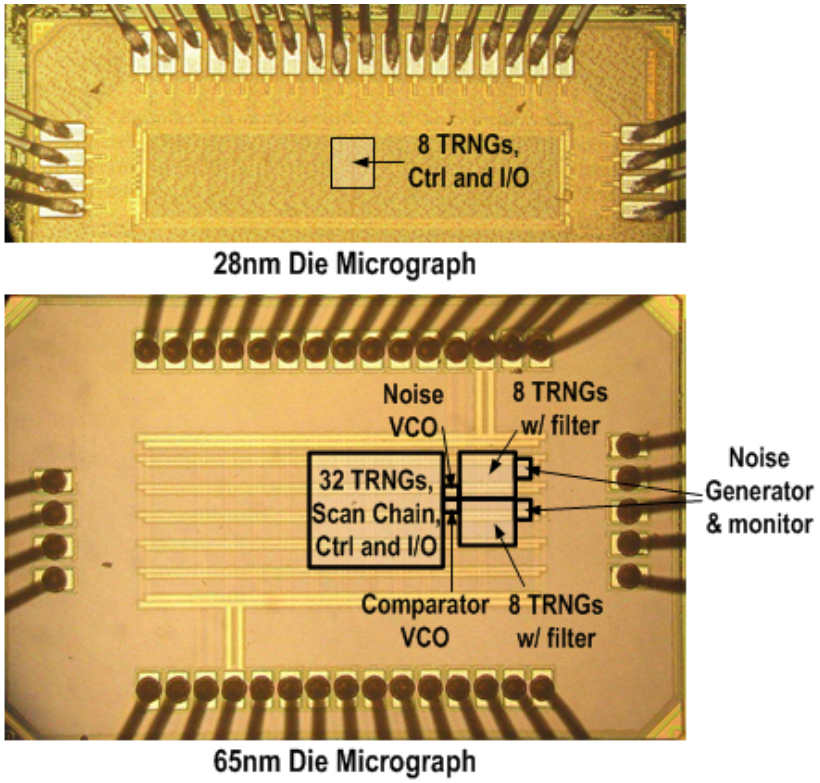


Figure 3.6: Die micrographs of 28nm and 65nm TRNG test chips.

Using an RF signal generator, up to 600mVpp noise is injected on the power supplies (after removing PCB decoupling caps) to test TRNG robustness against off-chip attack. The 65nm TRNGs retain randomness up to 360mVpp noise without filter and up to the 600mVpp generator limit with filter. To compensate for filter IR drop, TRNGs with filters operate at 5% increased supply voltage, incurring a slight power penalty. Since ROs in 28nm TRNGs operate at a higher frequency they are less sensitive to external attack; even unfiltered versions did not suffer randomness degradation

NIST Pub 800-22, rev. 1a, 2010 Randomness Tests	65nm, 21 stage RO, 0.9V, 2.80Mb/s		28nm, 21 stage RO, 0.9V, 23.16Mb/s	
	P-valueχ^2	Pass Rate	P-valueχ^2	Pass Rate
Frequency	0.785562	296/300	0.872947	297/300
Block Frequency	0.082177	297/300	0.746572	297/300
Cumulativ Sum	0.462245	294/300	0.955835	296/300
Cumulativ Sum	0.942895	295/300	0.329332	294/300
Runs	0.220931	296/300	0.574903	297/300
Longest Runs	0.329332	296/300	0.81047	298/300
Matrix Rank	0.046668	294/300	0.000682	296/300
FFT	0.03013	295/300	0.224821	295/300
Non Overlapping Template	PASS*	PASS*	PASS*	PASS*
Overlapping Template	0.878107	297/300	0.329332	296/300
Linear Complexity	0.487885	297/300	0.304126	295/300
Universal	0.935716	98/100	0.719747	99/100
Random Excursions	PASS*	PASS*	PASS*	PASS*
Random Excursions Variant	PASS*	PASS*	PASS*	PASS*
Approximate Entropy	0.514124	100/100	0.275709	100/100
Serial	0.304126	99/100	0.897763	99/100
Serial	0.867692	99/100	0.595549	100/100

* "PASS" means all sub tests pass minimum requirement.

** Minimum p-value χ^2 is 0.0001. Minimum pass rate is 291/300 for first 10 tests (using 300 × 40K bits) and 96/100 for the other 5 tests (using 100 × 1M bits).

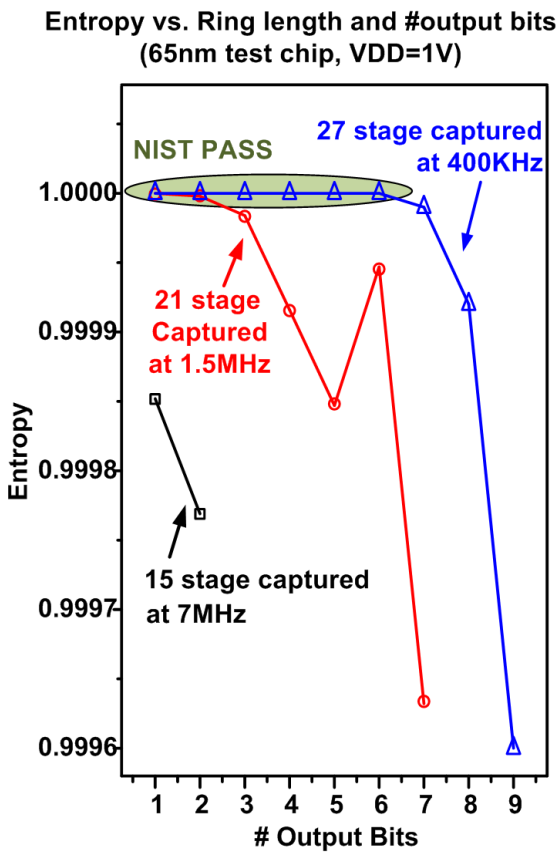


Figure 3.7: Measured NIST randomness test results and impacts of RO length and the number of harvested random bits on output data entropy.

at the generator limit. Electromagnetic interference also did not cause failure in any randomness tests.

Figure 3.8 shows the impact of supply noise on TRNG performance using on-chip noise generation. Even though a deliberate attacker will not have access to such a noise source, this test can demonstrate how readily a 3-edge TRNG can be integrated with noisy circuits on an SOC. TRNGs showed sensitivity to supply noise at frequencies near $1\times$ and $4\times$ nominal RO frequencies, reducing collapse time mean and variance. Randomness degrades at $> 125mV$ noise amplitude and $4\times$ frequency without a filter, but is recovered using a filter. Denial of service occurs when a TRNG cannot generate outputs due to external influence. This is observed only in unprotected TRNGs with on-chip noise at exactly $3\times$ nominal frequency since the ring locks to its $3\times$ frequency mode, preventing collapse. In this case, yield (% of master cycles that generate outputs bits) drops to

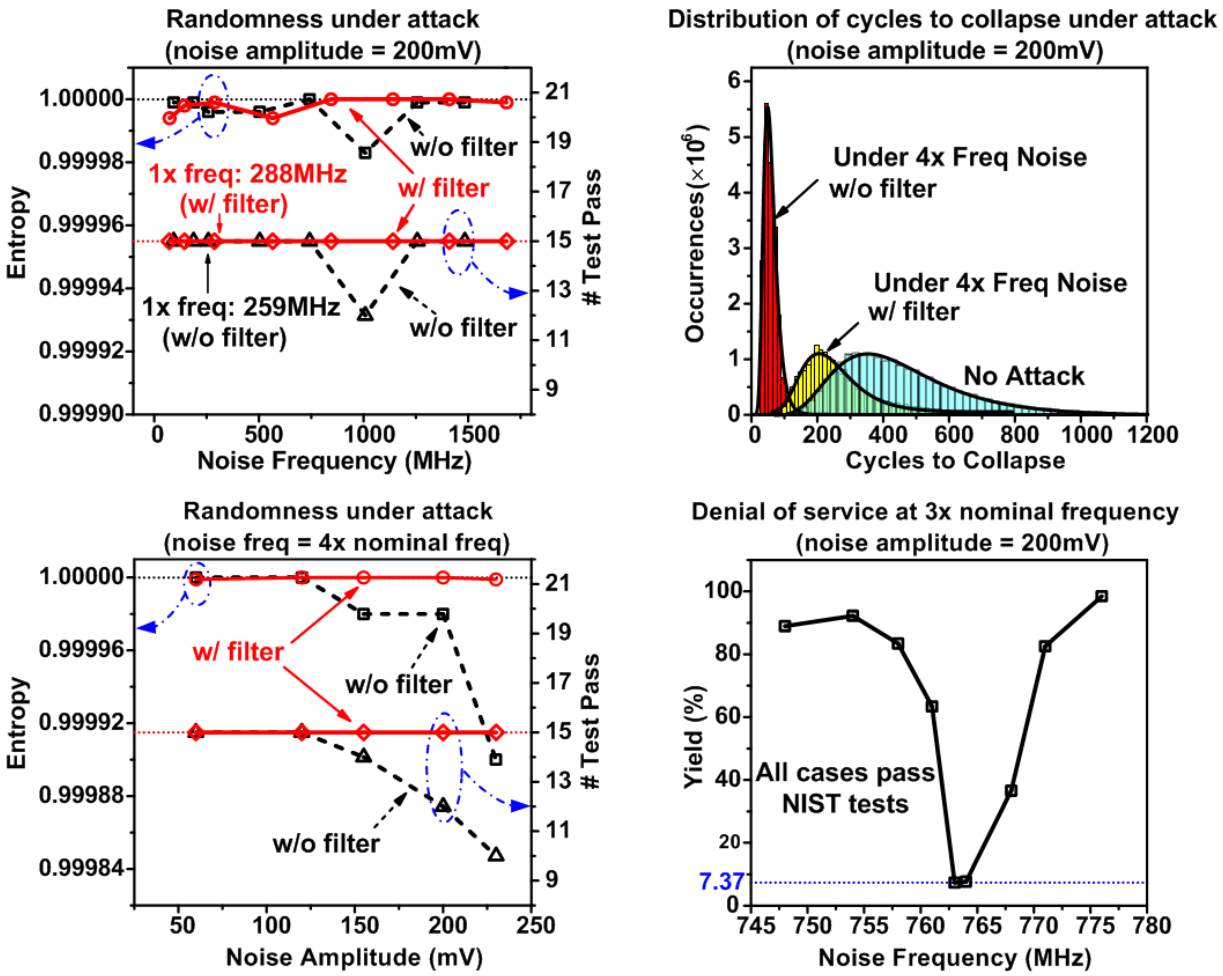


Figure 3.8: Measured impacts of on-chip noise frequency and amplitude on randomness of protected and unprotected TRNGs (65nm, 21-stage RO TRNG).

7.37%. Generated bits remain random (i.e. passing all NIST randomness tests). Table 3.1 summarizes measurement results with comparisons to prior work. The 28nm TRNG generates random bits at 23.16Mb/s while consuming 0.54mW and 375μm^2.

3.5 Summary

In summary, a synthesizable true random number generator constructed entirely using standard cell library and conventional placement & routing tools is proposed, which represents a "soft IP" TRNG that can pass all NIST randomness tests without pre calibration of post processing. Frequency collapse phenomenon in odd-stage ring oscillators is employed because of its immunity

Table 3.1: Summary of measurement results and a comparison with state-of-the-art hardware TRNG designs.

	This work (25°C, 0.9V core supply)		**JSSC 12 [10]**	**VLSI 11 [17]**	**ISSCC 08 [9]**	**JSSC 08 [11]**	**ISSCC 06 [8]**	**TC 03 [12]**
Technology	28nm	65nm	45nm	65nm	0.25µm	0.13µm	0.12µm	0.18µm
Entropy Source	Jitter in 3-edge RO		Metas-tability	Oxide breakdown	SiN MOS-FET Noise	Metas-tability	Oxide traps	Oscillator jitter
Design Method	Synthesized		Custom digital	Custom digital	Custom analog	Custom digital	Custom analog	All digital
Bit Rate (Mb/s)	23.16	2.8	2400	0.011	2	0.2	0.2	10
NIST Pass	All	All	All	All	not reported[b]	5	not reported	not reported[b]
TRNG Core Area (μm^2)	375	960 (1080[a])	4004	1200	1200	36300	9000	16000
Power (mW)	0.54	0.159	7	2	1.9	1	0.05	2.3
Efficiency (nJ/bit)	0.023	0.057	0.0029	181.81	0.95	5	0.25	0.23
Post Processing	No	No	No	No	Yes	No	Yes	no
Resistance to Attack	Yes	Yes	Not reported	Not reported	Not reported	Not reported	Not reported	No[c]

a Including 1/8th of filter area (MIM cap and poly resistor), filter is shared by 8 TRNG here and placed beneath the MIM cap.
b Only the given number of NIST test results are reported
c Only NIST FIPS 140-2 test result is provided, which is out dated, less strict and requires only 20,000 bits compared to NIST Pub 800-22

to process variations. Since no calibration circuits is required, the design achieves a minimized area of $375\mu m^2$ in 28nm technology. The low complexity and area features of this design make it appealing to low-cost devices.

CHAPTER 4

Physically Unclonable Function for Chip Identification and Secret Key Storage

4.1 Introduction

Physically Unclonable Functions (PUFs) are among the most promising security primitives as low-cost solutions for key storage, chip authentication, and semiconductor supply chain protection. A PUF is a function that maps an input code ("challenge") to an output code ("response") in a manner that is decided by random process variations and therefore unique for every chip. Two types of PUFs exist in literature: a "strong" PUF with a large challenge-response space [22, 23] and a "weak" PUF providing a limited length key (also known as chip ID) [24–28]. Theoretically, "strong" PUF can be used for a wider range of applications, such as key generation and device authentication, but a "weak" PUF is still appealing to many applications because of its guaranteed randomness and better reproducibility. A new "weak" PUF design is described in this chapter and a robust "strong" PUF design is shown in Chapter 5.

Weak PUFs typically have an array of identically designed PUF cells that leverage device mismatch in fabrication as static entropy source, and serve as a low-cost and more secure alternative to non-volatile-memory-based key storage. Output reproducibility across PVT variations and density of the array are two critical metrics directly related to the security and cost of a PUF. Recent works have presented custom PUFs based on NAND gates [24], current mirrors [25], PTAT [26], and cross-coupled inverters [27, 28]. These designs outperform conventional SRAM-based PUFs,

but all sacrifice some metrics comparing to each other. For example, [25, 27] are large, [26, 28] has lower native stability and energy efficiency, while [24] is sensitive to supply voltage and may experience large short circuit current. Finally, IoT and wireless sensor nodes tend to use older technologies for lower cost and standby power, which is challenging for PUF design because of smaller process variations.

This work presents a PUF cell based on a simple sub-threshold 2-transistor (2T) amplifier implemented in 180nm CMOS featuring: (1) a small $553F^2$ PUF cell, integrated in an array with all peripheral circuits; (2) excellent stability: 1.65% native unstable bits, reaching 0.05% unstable bits with 11b temporal majority voting (TMV), and 3.16% and 2.01% flipping bits across wide temperature (-40-120$^{\circ}C$) and voltage (0.8-1.8V) ranges; (3) high energy efficiency of 11.3fJ/b at nominal 1.2V and 1.5fJ/b at 0.8V; (4) high throughput (4.8Gb/s) via highly parallel operation, despite using an older technology. A masking technique using body bias is employed to find unstable bits without costly temperature sweeps. The design and implementation details of the PUF is described in Section 4.2; measurement results of 180nm prototype chip are included in Section 4.3; and finally, the work is summarized and compared with previous works in Section 4.4.

4.2 Weak PUF Cell using 2-Transistor Amplifiers

The key to improve reproducibility and uniqueness of PUF responses is to have static operation and in-cell amplification and quantization, because the impacts of noise and variation of shared quantizers can be minimized. A PUF cell based on a sub-threshold 2-transistor (2T) structure acting in two different ways (an amplifier and a voltage generator) is proposed following these guidelines with minimum overhead.

In this 2T structure shown in Figure 4.1, standard-V_{th} NMOS with $V_{gs} = 0$ sets a sub-threshold current. When the PMOS gate is used as an input port, the 2T structure is a common-source amplifier with pseudo-resistor loading. Thanks to the large gm in sub-threshold, gain larger than 40 is provided by minimum-sized transistors. When the input and output of the amplifier are shorted as in Figure 4.2, the circuit generates an output voltage that is equal to the switching voltage of the amplifier. The voltage can be calculated as below.

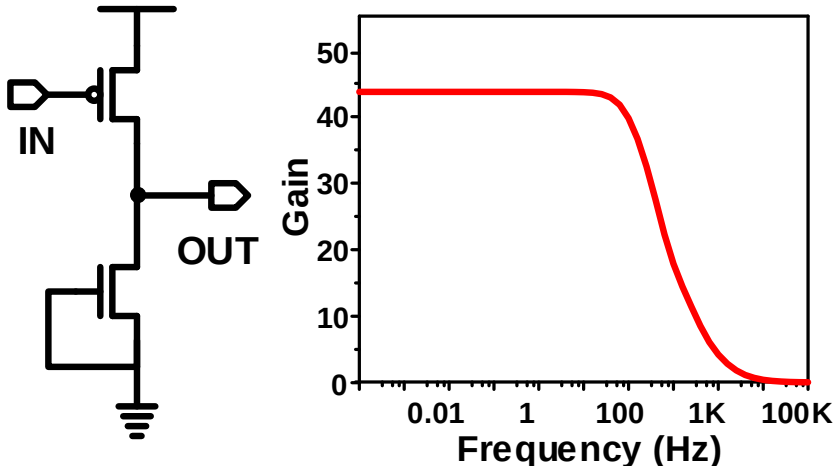


Figure 4.1: 2-transistor sub-threshold amplifier.

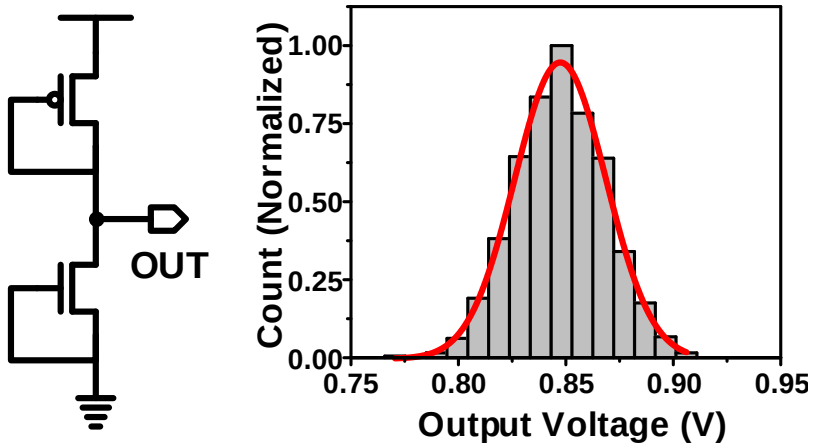
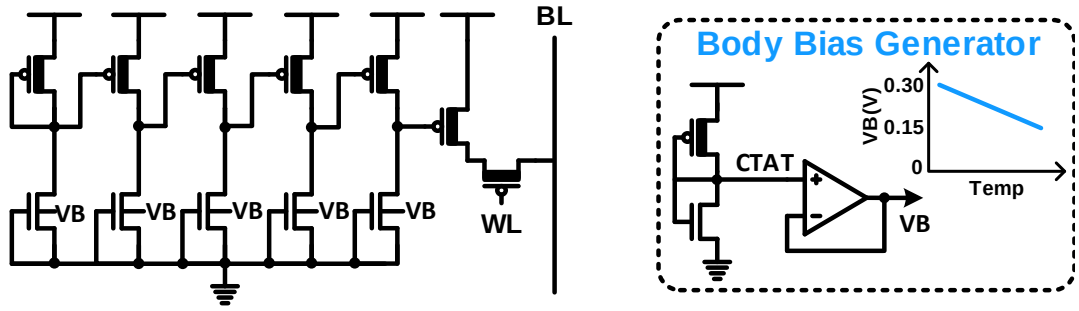


Figure 4.2: 2-Transistor "switching" voltage generator provides the random sources for PUF.

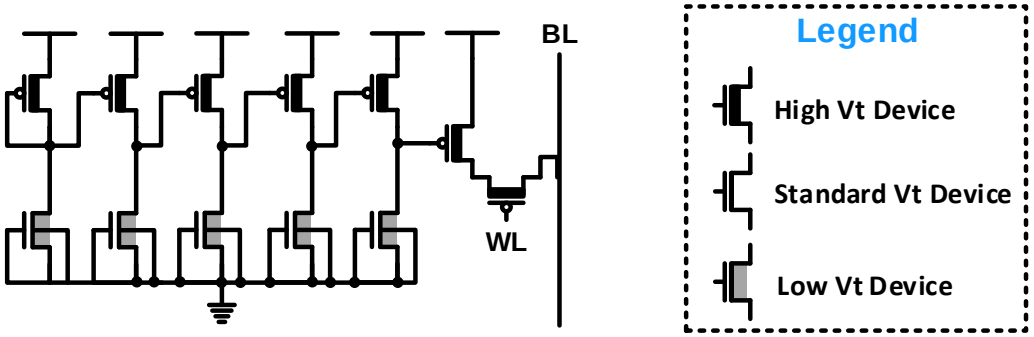
$$\mu_n C_{ox,n}(W/L)_n exp\left(\frac{0-V_{th,n}}{kT/q}\right) = \mu_p C_{ox,p}(W/L)_p exp\left(\frac{V_{dd}-V_{out}-|V_{th,p}|}{kT/q}\right) \tag{4.1}$$

$$V_{out} = V_{dd} + V_{th,n} - |V_{th,p}| + \frac{kT}{q} ln\left(\frac{\mu_p C_{ox,p}(W/L)_p}{\mu_n C_{ox,n}(W/L)_n}\right) \tag{4.2}$$

This "switching" voltage tracks VDD and solely depends on the threshold voltage differences between top and bottom devices, assuming both devices have $|V_{ds}| > 200mV$, allowing sub-threshold drain-current dependence on V_{ds} to be ignored. When a 2T amplifier is connected to the output of an identically sized 2T voltage generator with the same switching voltage (neglecting mismatch), the amplifier output voltage equals its input voltage. However, mismatch will induce a small difference in the switching voltages of the 2 structures, which will then be amplified by the



(a) DNW (deep n-well) Version PUF cell with CTAT body bias



(b) LVT (low Vth) Version PUF cell

Figure 4.3: Schematics of two implementations of 2T PUF.

large amplifier gain. The switching voltage follows a normal distribution (Figure 4.2) and therefore the difference also follows a normal distribution. Four amplifier stages are employed to amplify the voltage difference to full rail in $> 99.9\%$ cases. This full-rail signal is then used as the digitized PUF output.

To implement the design in CMOS and satisfy the requirements on V_{ds}, the two transistors must have enough difference in their threshold voltages. We proposed two implementations shown in Figure 4.4, one is based on body biasing the NMOS to lower its threshold voltage, and the other one is directly using low V_{th} NMOS. A complementary-to-absolute-temperature bias generator is designed to provide body bias for the DNW version and found to be beneficial to the reproducibility of the PUF outputs. By embedding high gain amplification in the PUF cell, we significantly improve PUF's reproducibility and reduce its design complexity compared to shared amplifiers with noise and offset cancellation. It also enables arrangement of the PUF cells in a 16 by 64 crossbar array (Figure 4.4). This improves PUF density and energy efficiency compared to using scan chains [25, 27] or multiplexers [24] to read the PUF outputs. Another advantage of the

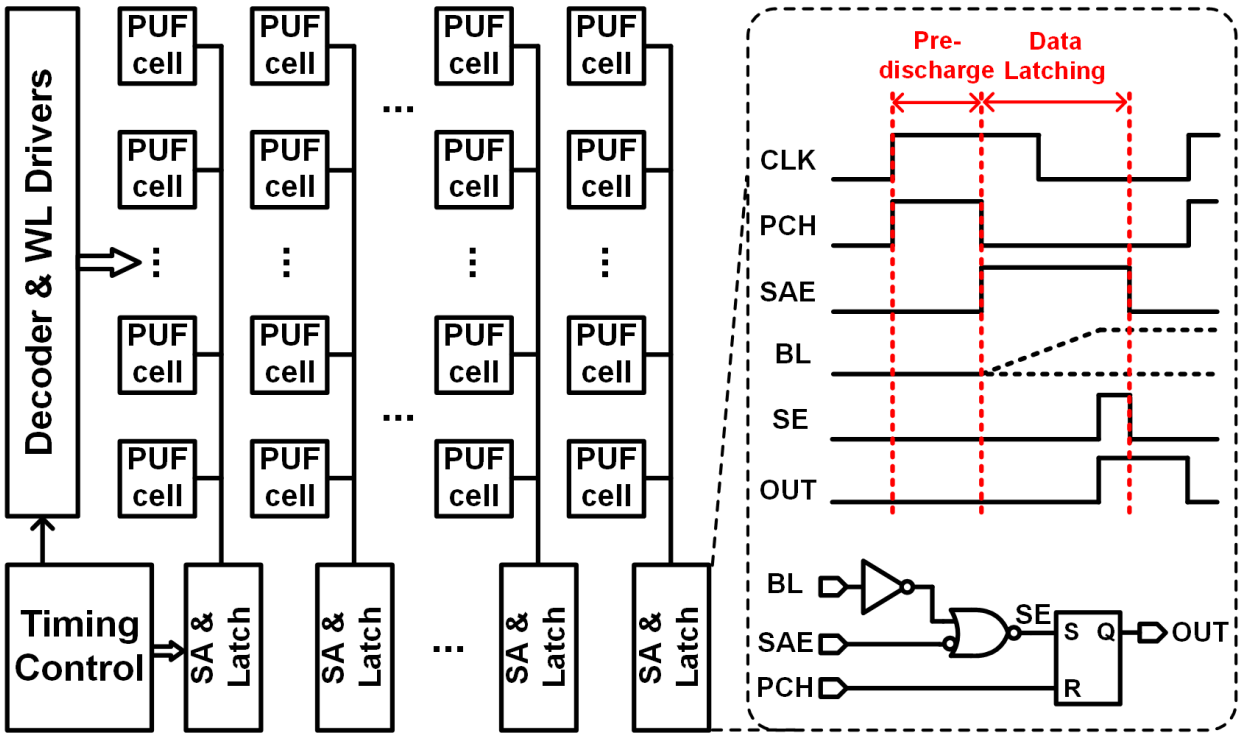


Figure 4.4: PUF crossbar array diagram along with read out circuits and waveforms, which are similar to that of SRAM.

crossbar configuration is that most SRAM read-assist techniques can be applied to improve read robustness and performance.

4.3 Measurement Results

The proposed PUF is implemented in 180nm CMOS for ultra-low power consumption and to prove that the design works in a mature technology with less process variations. A die photo along with PUF cell layouts are shown in Figure 4.5. To fully characterize the PUF across process, we tested both typical corner (TT) dies and intentionally skewed corner dies (FF, SS, FS, SF).

Uniqueness between PUF instances and uniformity within PUF array are fundamental requirements for PUFs. The Hamming Distance (HD) between two PUF response words of equal length provides a measure for uniqueness, which calculates the number of positions at which the corresponding symbols are different. Figure 4.6 shows both inter-die Hamming Distance (HD) characterizing the uniqueness between chips and intra-die HD charactering temporal stability of the output. Ideally, inter-die HD should have a normalized average value of 0.5, which indicates max-

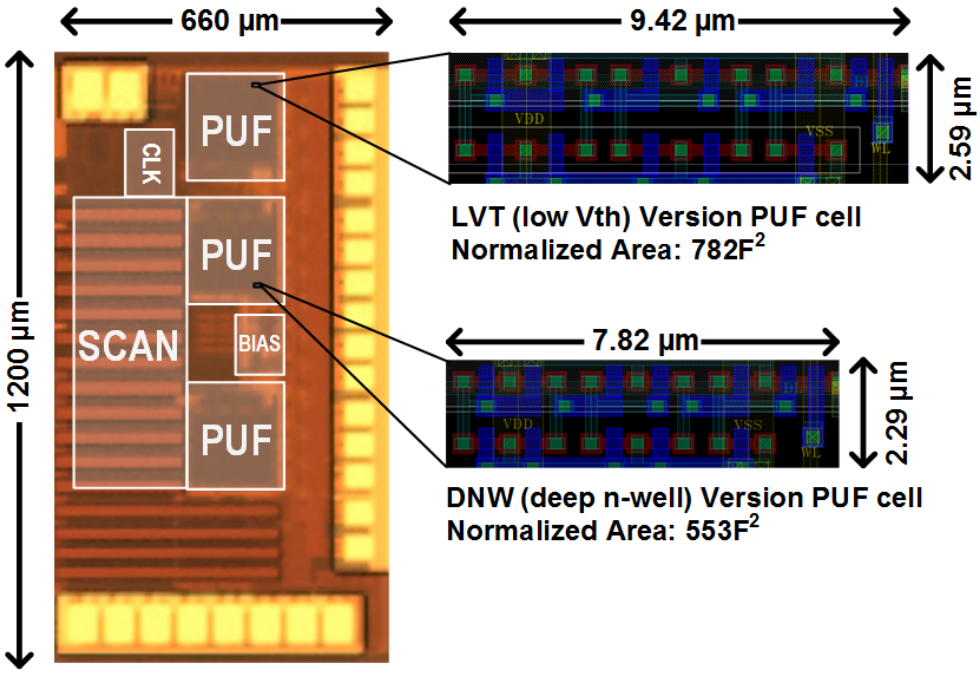


Figure 4.5: Die micrograph of 180nm PUF test chip and PUF cell layouts.

imum uniqueness. As can be seen, both DNW and LVT versions of PUF exhibits close to ideal HD distributions. Intra-die HD shows only 0.0008 and 0.0007 bit differences for the 2 PUF versions, providing $> 600/700\times$ separation between inter- and intra-die average HDs (identifiability of PUF), which has been significantly improved compared to the previous best reported value of $143\times$ [25]. Uniformity among PUF cells inside the same array is demonstrated by spatial autocorrelations. Measurements in Figure 4.6 confirms that autocorrelation is bounded to 0.0173 and 0.0167 with 95% confidence level for the two implemented PUF versions.

PUF output reproducibility across process, voltage and temperature (PVT) variations is the most critical metric for PUFs. Several measurement methods exist in literature. Bit error rates (BERs) and percentage unstable bits (♯ of bit locations with at least one error across all evaluations) are common metrics for nominal condition reproducibility [27]. BER is a direct measure of noise effects while percentage of unstable bits is a stricter metric affected by number of samples. For temperature and voltage sweeping results, BER and percentage of flipping bits (♯ of bit locations that give flipped results even after long enough measurements) are most interesting metrics to consider [26], the former one includes both V/T and noise effects while the latter one considers only V/T effects. In this work, we measure all these metrics for a fair comparison with prior arts

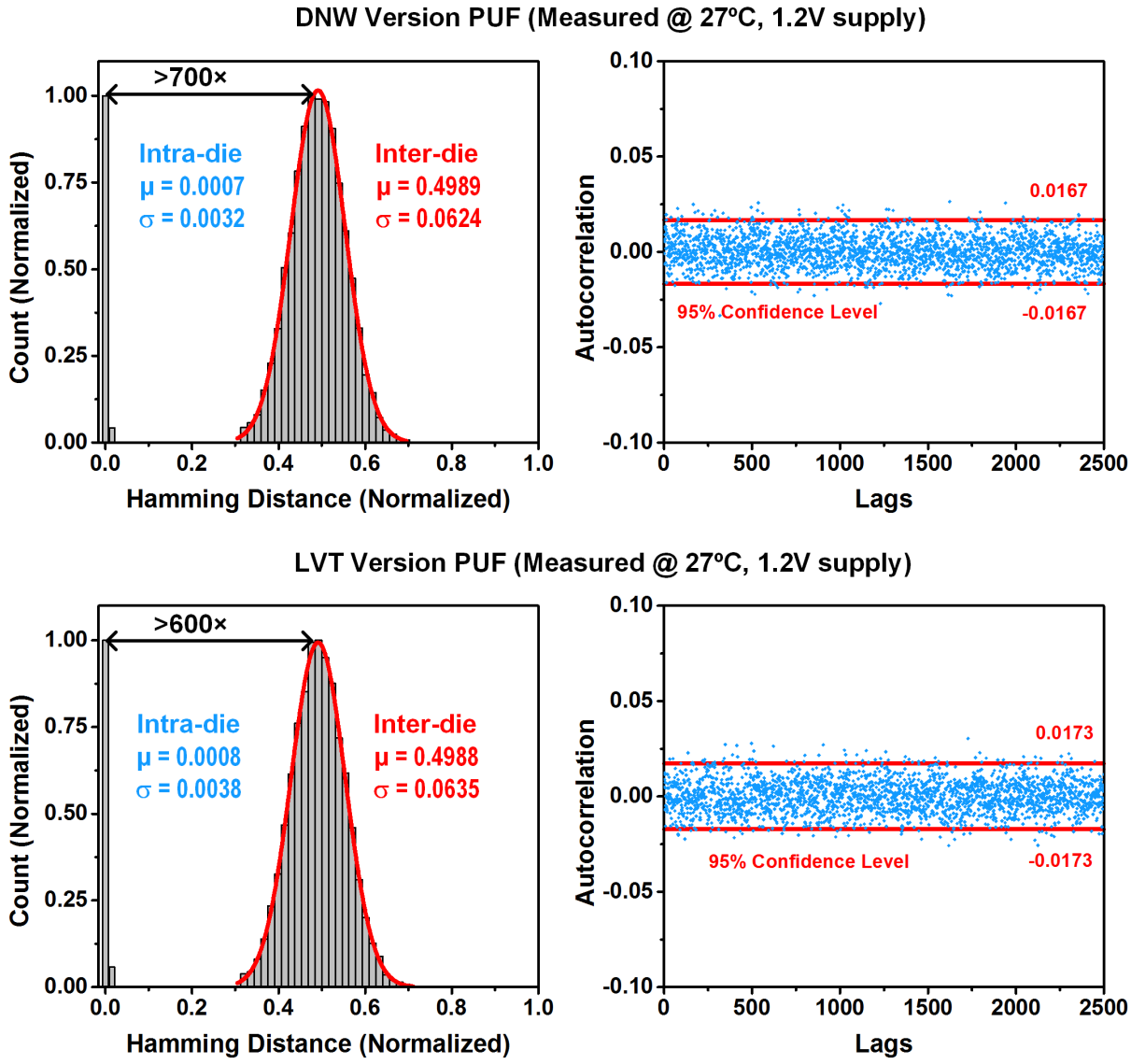


Figure 4.6: Measured intra-die/inter-die Hamming Distances, spatial autocorrelation function across 14 chips (6TT, 2FF, 2SS, 2FS and 2SF).

and also for studying the differences between their results. The measurement steps we take are shown in Figure 4.7.

Reproducibility results at nominal condition is shown in Figure 4.8. Figure 4.8a and 4.8b show the results of PUF with deep n-well and body bias, while Figure 4.8c and 4.8d plot the results of PUF using low V_{th} NMOS transistors. From the comparison of the two metrics, it is clear that the BER results stabilize after a few hundred evaluations while the percentage of unstable bits plateaus after thousands of evaluations. This is expected because the latter metric consider more of

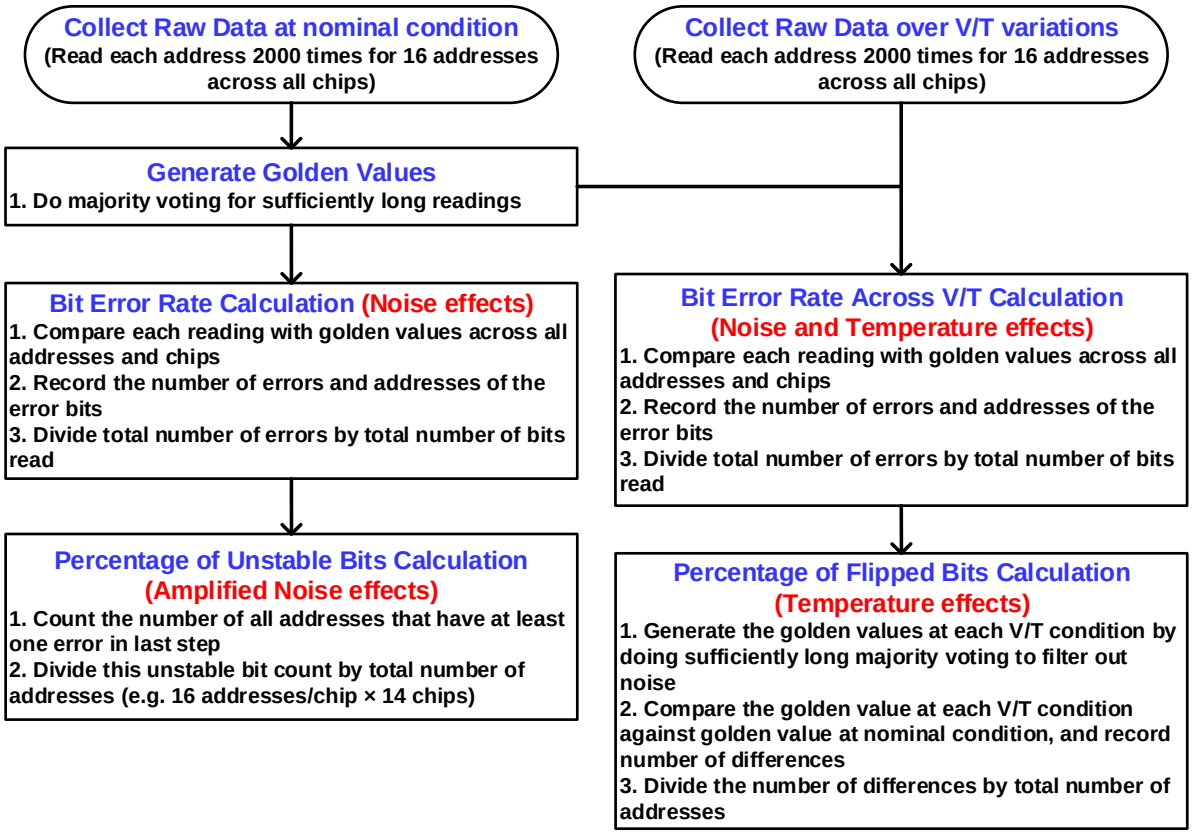
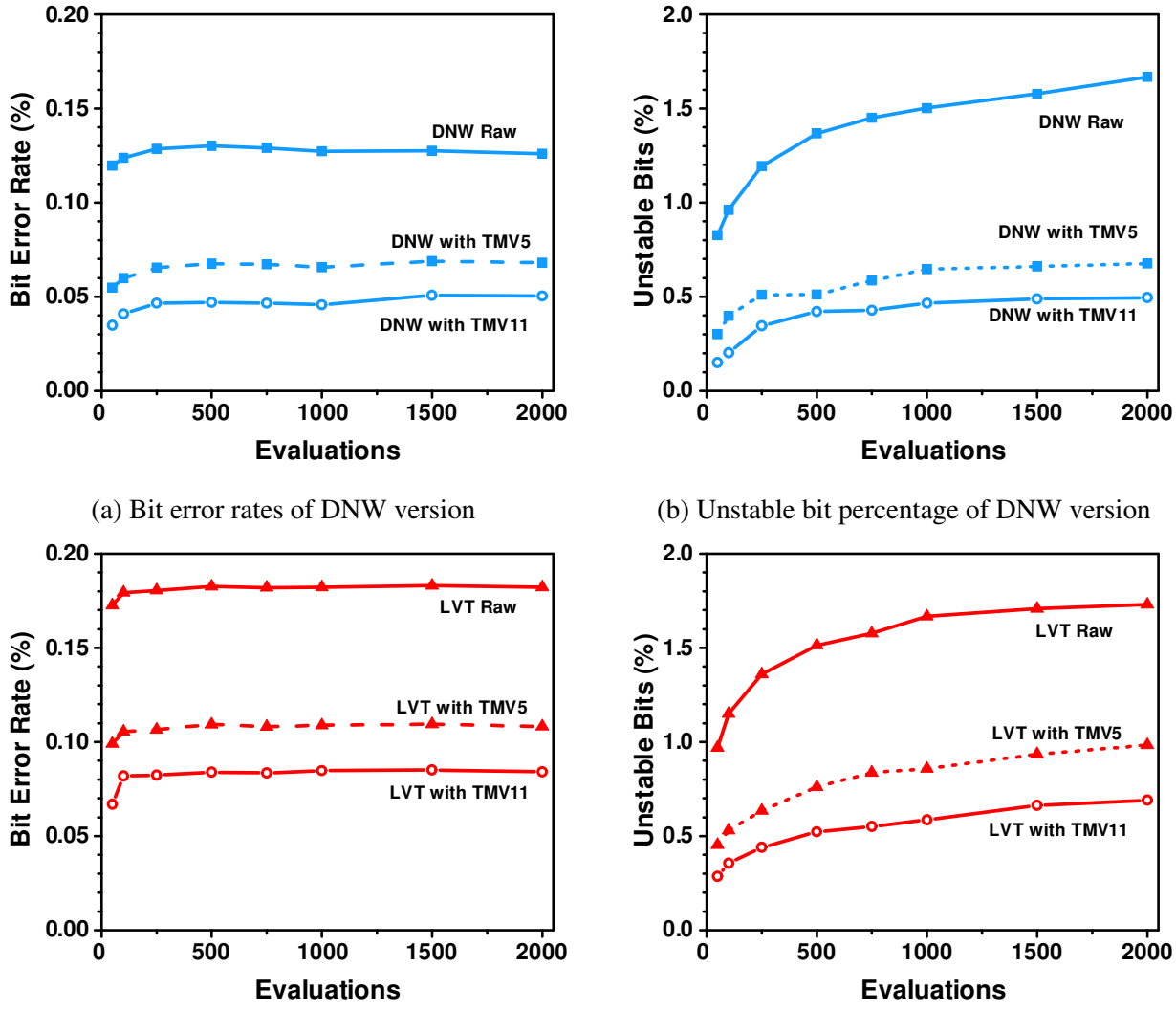


Figure 4.7: Measurement steps for various PUF reproducibility metrics.

the extreme cases in statistical measurements. As shown in all measurements results in Figure 4.8, the reproducibility of the 2T PUF can be further improved with 5-bit or 11-bit temporal majority voting (TMV), which means taking 5 or 11 consecutive measurements and do a majority voting of all measurements to filter out noise. The benefits of TMV saturates fast with number of bits and 11 is an optimal number for 2T PUF considering the trade-off between reproducibility and energy.

Measurement results of bit reproducibility across V/T variations are illustrated in Figure 4.9 and Figure 4.10. Average results of all corner chips show as few as 0.2% additional flipping bits per 10°C change and 0.2% per 0.1V change across 40-120°C and 0.8-1.8V. The DNW version flipping bit data at 4 corners is also plotted to show that stability is not significantly affected by global process variation.

As shown in Figure 4.11, the 180nm PUF delivers 4.8Gb/s with 64b wide outputs, thanks to the crossbar array, which can improve start-up time for duty-cycled IoT systems. The two versions of



(a) Bit error rates of DNW version

(b) Unstable bit percentage of DNW version

(c) Bit error rates of LVT version

(d) Unstable bit percentage of LVT version

Figure 4.8: Measured bit error rates (BER) and percentage of unstable bits over ♯ of PUF readings, with and without temporal majority voting (TMV) at nominal 1.2V supply voltage and 27°C.

PUF core consume 11.3fJ/b and 13.5fJ/b, respectively. This number includes PUF cell static power, bias generator/analog buffer static power, and bitline driver dynamic power. Using this energy measurement allows a fair comparison with prior works using scan chain to read PUFs. Total PUF cell static power is 26pW per cell while the shared bias generator/analog buffer consumes a total of 185pW. Total PUF array energy including all peripherals (timing generation, decoder, WL driver, and BL latches) is 91.1fJ/b. The PUF functions across 0.8 to 1.8V power supply because further lowering V_{DD} violates the assumption of large V_{ds} and make the reproducibility worse. From energy efficiency measurement results in Figure 4.12, the optimal efficiency occurs

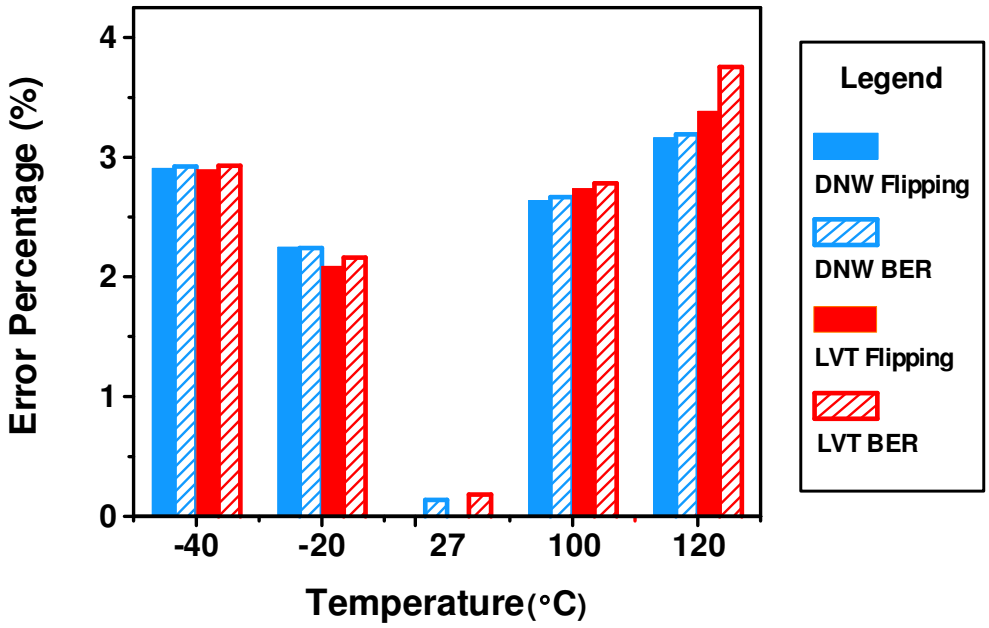


Figure 4.9: Measured bit error rates and percentage of flipping bits across temperature variations.

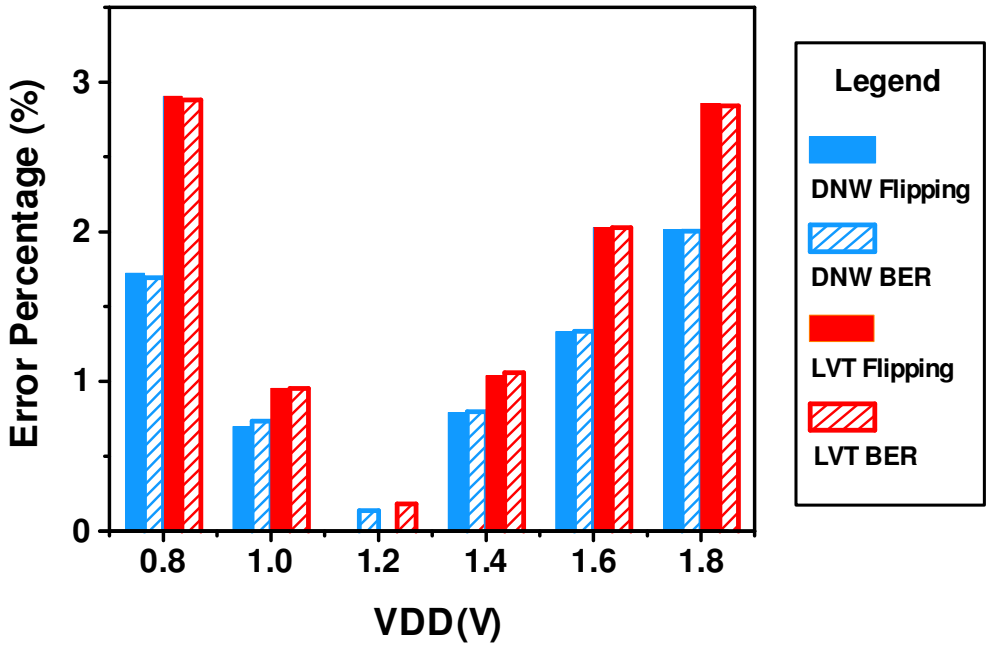


Figure 4.10: Measured bit error rates and percentage of flipping bits across V_{DD} variations.

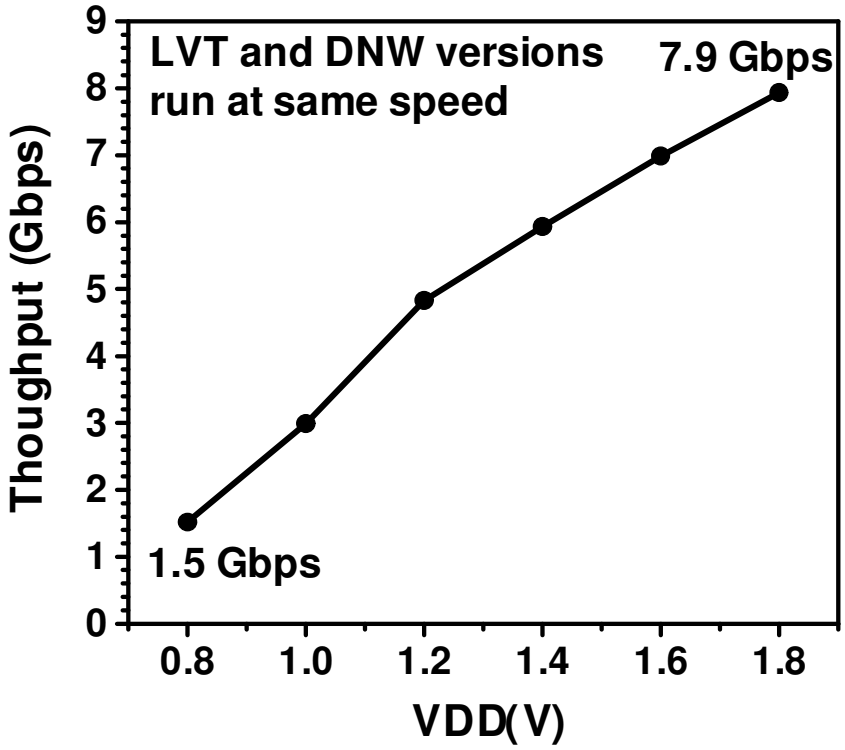


Figure 4.11: Measured PUF throughput across V_{DD}.

at 0.8V with 1.5fJ/b core power and 52fJ/b total power for the DNW version. Table 4.1 summarizes measurement results for both PUF versions and compares them to state-of-the-art "weak" PUFs.

To further improve reproducibility, masking technique is commonly employed to filter out unstable bits (dark bits). Conventional approaches include: 1) finding unstable bits by many evaluations at room temperature [27], which often misses bit flips due to temperature variations; and 2) finding unstable bits by sweeping temperature [24], which incurs high testing cost. This work uses external control of the PMOS n-well voltage (connected to V_{DD} during normal operation) to generate threshold voltage shifts and emulate temperature changes. The PMOS threshold voltage affects both amplifier gain and the switching voltage, and as the input transistor, offers a larger impact than NMOS body biasing. The n-well body bias is swept at room temperature, avoiding the high costs of actual temperature sweeping. The percentage of bits that are marked as unstable at room temperature is shown in Figure 4.13, which increases as body biasing become stronger. Without n-well body biasing, masking only improves BER at -40 and 120$^{\circ}C$ by 15.6% compared to no masking. As shown in Figure 4.14, the new body biasing technique improves BER by up to

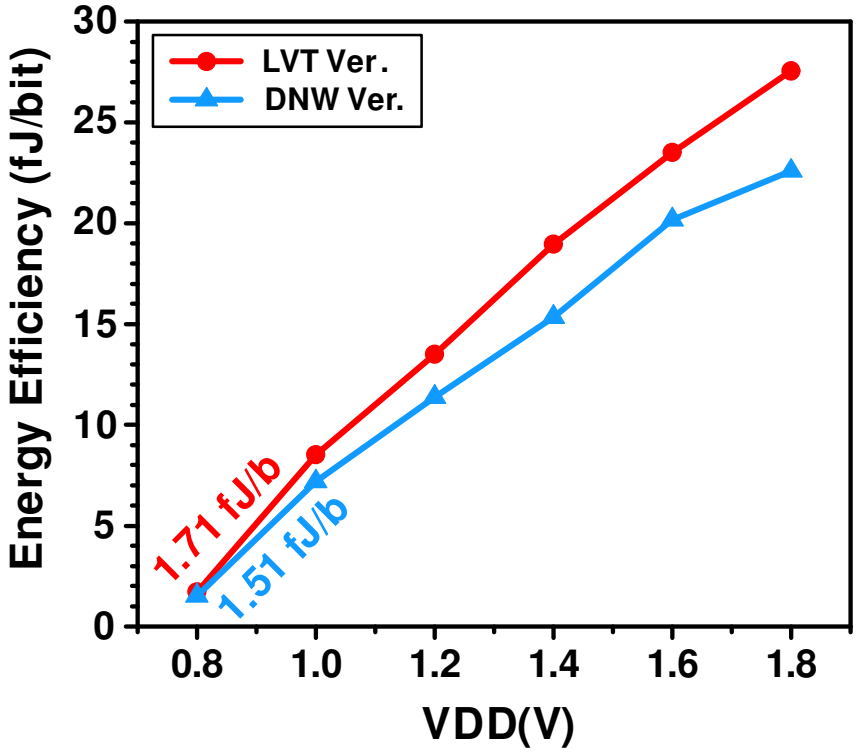


Figure 4.12: Measured PUF efficiency across V_{DD}.

60% with ±0.3V body bias applied during initial room temperature testing.

4.4 Summary

As shown in Table 4.1, the proposed 2T PUF breaks the conventional trade-off between reproducibility, energy and area and achieves optimal all-around performance. The sub-threshold 2T structure is employed as both process variation generator and amplifier, which removes the extra power and area cost of shared quantizers and achieves optimal noise tolerance and reproducibility across V/T variations. The design can be readily used for chip identification and secret key storage applications. Its low-power and high-efficiency operation makes it suitable for IoT applications.

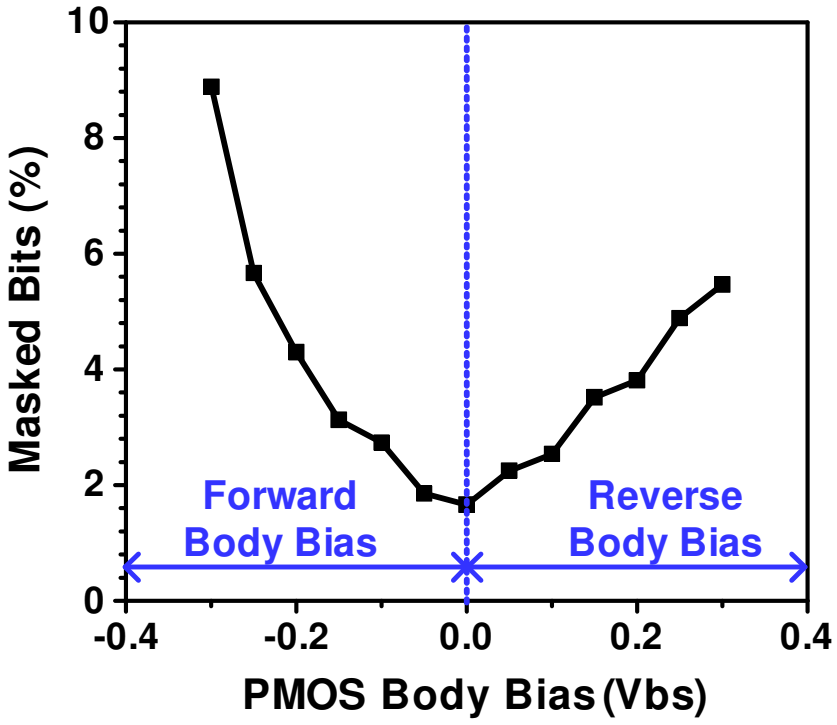


Figure 4.13: Percentage of masked bits at room temperature.

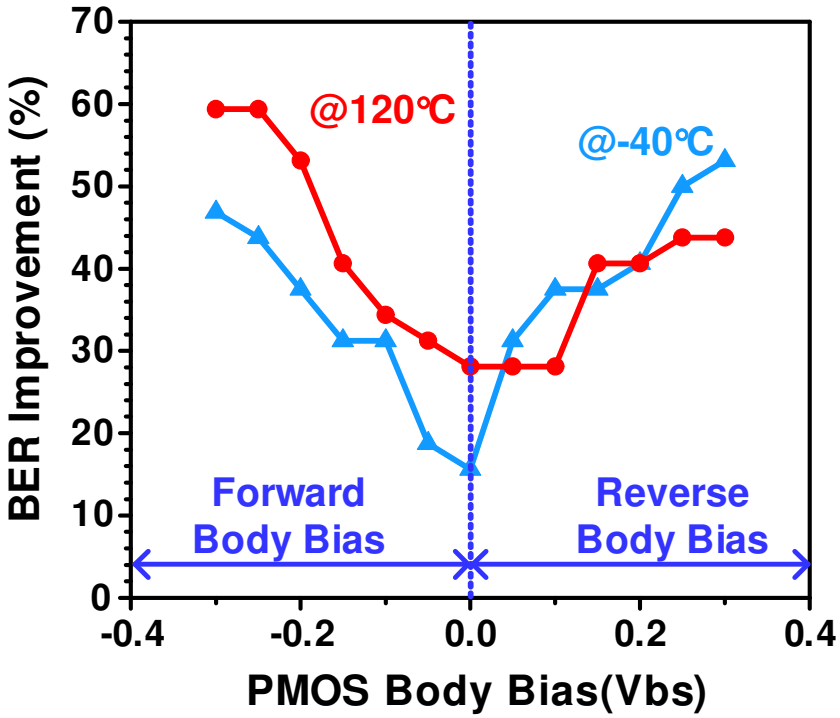


Figure 4.14: Bit error rates improvement at extreme temperatures as a function body bias applied during initial testing at room temperature.

Table 4.1: Summary of measurement results and a comparison with state-of-the-art silicon weak PUFs.

		This work (LVT Version)	This work (DNW Version)	ISSCC 16 [24]	ISSCC 15 [25]	VLSI 15 [26]	ISSCC 14 [27]	JSSC 08 [28]
Technology		**180nm**		45nm	65nm	65nm	22nm	130nm
PUF Cell Area/Bit (F^2)		**782**	**553**	2613	6036	756	9628	1092
Total Area/Bit (F^2)		**1082**	**843**	-	~36450	1756	-	1767
Native Unstable Bits (# of evaluations)		**1.73% (2000)**	**1.67% (2000)**	-	1.73% (400)	6.54% (500)	30% (5000)	-
Native Unstable Bits (# of Majority voting)		**0.69% (TMV 11)**	**0.50% (TMV 11)**	-	-	2% (TMV 11)	3%[a] (TMV 15)	-
Bit Error Rates (nominal condition)		**0.18% 0.08% (TMV 11)**	**0.13% 0.05% (TMV 11)**	0.1% [a]	-	-	8.3% 0.97% [d]	3.04%
Tested Operating Conditions	**Temp (°C)**	**-40~120**		-25~85	25~85	0~80	25~50	0~80
	Supply (V)	**0.8~1.8**		-	0.7~1	0.6~1.2	0.7~0.9	0.9~1.2
Bit Errors per 10°C		**0.21%**	**0.2%**	0.15%	0.47%	0.44% [b]	-	0.68%
Bit Errors per 0.1V		**0.29%**	**0.2%**	-	1.27%	0.17% [c]	0.49% [d]	1.82%
Bit Rate (Mb/s)		**4832 @1.2V**	**4832 @1.2V**	1.92	-	10.2	2000	1
PUF Core Energy (fJ/bit)		**13.5 @1.2V 1.71 @0.8V**	**11.3 @1.2V 1.51 @0.8V**	-	15	548	13	930
Norm. Inter-PUF Hamming Distance		**0.499**	**0.499**	0.498	0.5014	0.5001	~0.49	0.506

a. With 2-bit glitch detection
b. Comparator is re-calibrated manually at each temperature
c. Using off-chip ADC
d. After burn-in, 15-bit temporal majority voting

CHAPTER 5

Physically Unclonable Function for Robust Chip Authentication

5.1 Introduction

Chapter 4 introduces the concepts of physically unclonalbe functions (PUFs) and two types of them: "strong" and "weak" PUFs. In this chapter, a "strong" PUF design is proposed as a secure method for chip authentication in insecure environments [23, 29, 30].

Conventional authentication methods using secret keys, digital signatures, and encryption have high computational expense and are vulnerable to tampering attacks[30]. PUFs address this by generating their secret, unique challenge/response pairs (CRPs) using process variation, thus eliminating the expense of programming a secret key and the risk of compromising the stored data [23, 29, 30]. In addition, PUFs rely on hardware intrinsic variations to provide one-way function, which is completely different from mathematical hard-to-solve problems and ideally can be more difficult to attack. A basic PUF-based authentication protocol consists of two phases as illustrated in Figure 5.1. Enrollment happens in a secure environment where a known chip (♯2435 in our example) is interrogated with a large number of random challenges and the resulting CRPs are stored. After the device is deployed in an untrusted environment, a chip claiming to be ♯2435 is interrogated with a small subset of the stored challenges and the chips response is verified against golden values. If Chip ♯2435 was swapped with a fraudulent one, its response would not match and authentication would fail. It is critical that each verification attempt uses a new subset of stored challenges. Hence, illicit observation of previously used CRPs is harmless since each CRP is used only once. This is a key differentiator from "weak" PUFs, or chip IDs, which are considered PUFs

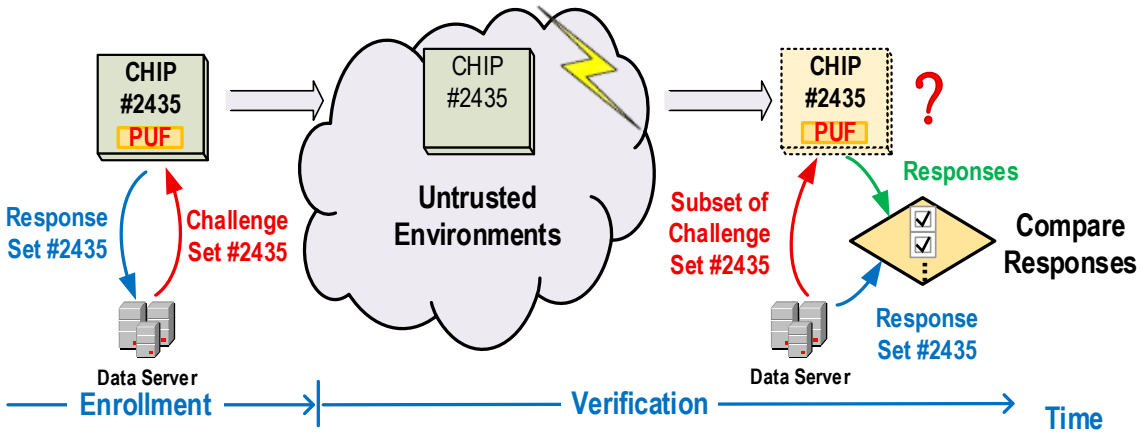


Figure 5.1: Basic PUF authentication protocol for resource-constrained devices.

with only a limited CRP space. In such a case any observation in an untrusted environment of Chip ♯2435's chip ID allows a malicious chip to impersonate Chip ♯2435 by storing and reporting the observed PUF response (chip ID).

Silicon PUFs have been proposed based on variations in gate and interconnect delay [31], ring oscillator frequency [14], and inverter maximum gain point [27]. Due to the sensitivity of device parameters to operating conditions (temperature, voltage, wearout), the PUF output may change between interrogations, manifesting as BER and possible authentication failures. Stabilizing approaches to address this includes majority voting, burn-in, ECC, and masking [23, 27, 29, 30]. However, these all require additional testing and calibration efforts for each chip.

This work presents a PUF based on multi-edge oscillation collapse in a ring-oscillator (RO). The PUF is validated in 40nm CMOS, featuring: (1) BER remains $< 10^{-8}$ across -25 to 125°C and 0.7 to 1.2V; (2) Average inter-chip hamming distance is 0.5007; (3) The all-digital design occupies 845μm^2 and requires no calibration. In this chapter, the concept and implementation of using frequency collapse in an even-stage RO for "strong" PUF is described in Section 5.2; measurement results of 40nm test chip are provided in Section 5.3; and the work is summarized in Section 5.4.

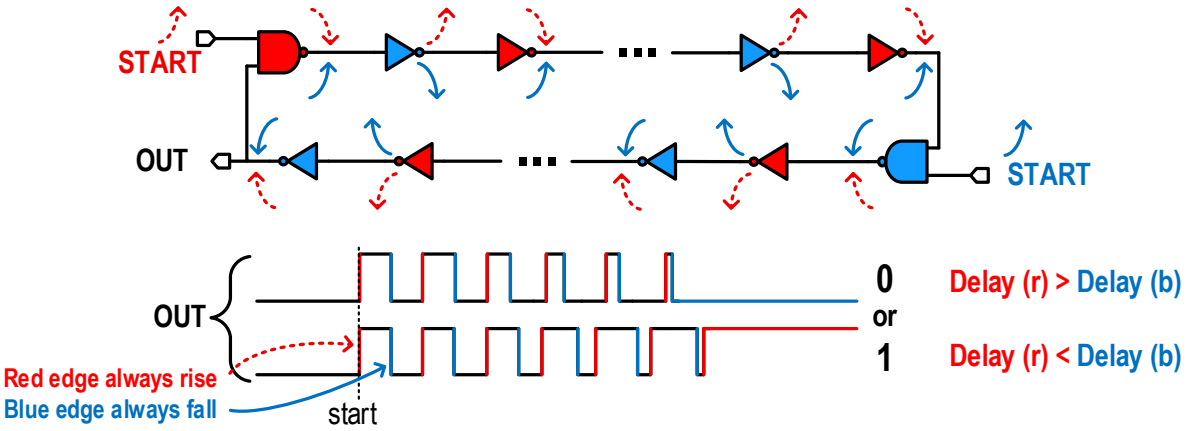


Figure 5.2: Robust PUF based on frequency collapse in even-stage ring oscillator.

5.2 Frequency Collapse based Physically Unclonable Function

The proposed PUF design translates physical variation to a digital output by injecting 2 edges into an even-stage RO (Figure 5.2). The two injected edges travel entirely different paths and hence accumulate delay cell mismatch, causing one edge to overtake the other and collapsing the oscillation. Depending on which path is faster, the output settles to either 0 or 1. Noise averaging along the paths of the two edges aids stability, which is further enhanced by simple dynamic thresholding based on cycles to collapse.

For the proposed structure to respond to a large set of challenges, each stage in the RO is selected from 8 identical delay cells (Figure 5.3). By selecting from 8 cells in each stage (3 bits of the challenge), rather than 2 cells (1 bit), the RO length is shortened, making the difference between the 2 paths larger (less averaging) and the output value more robust. To further increase response stability, we add a footer to each delay cell and bias it in near-threshold to ensure that the variation of the footer NMOS dominates the total delay. A CTAT is used to generate the bias voltage on-chip, which performs a first-order temperature compensation of the footer current to reduce the PUF temperature sensitivity. The RO and control logic is reset during the positive phase of the clock (CLK) with the PUF output generated during the negative phase.

Bit stability is an essential property for a reliable PUF. Modeling and measurement reveal that the number of cycles to collapse follows an inverse Gaussian distribution (see Chapter 2). Responses generated by slower frequency collapse have much larger average BER. This is expected

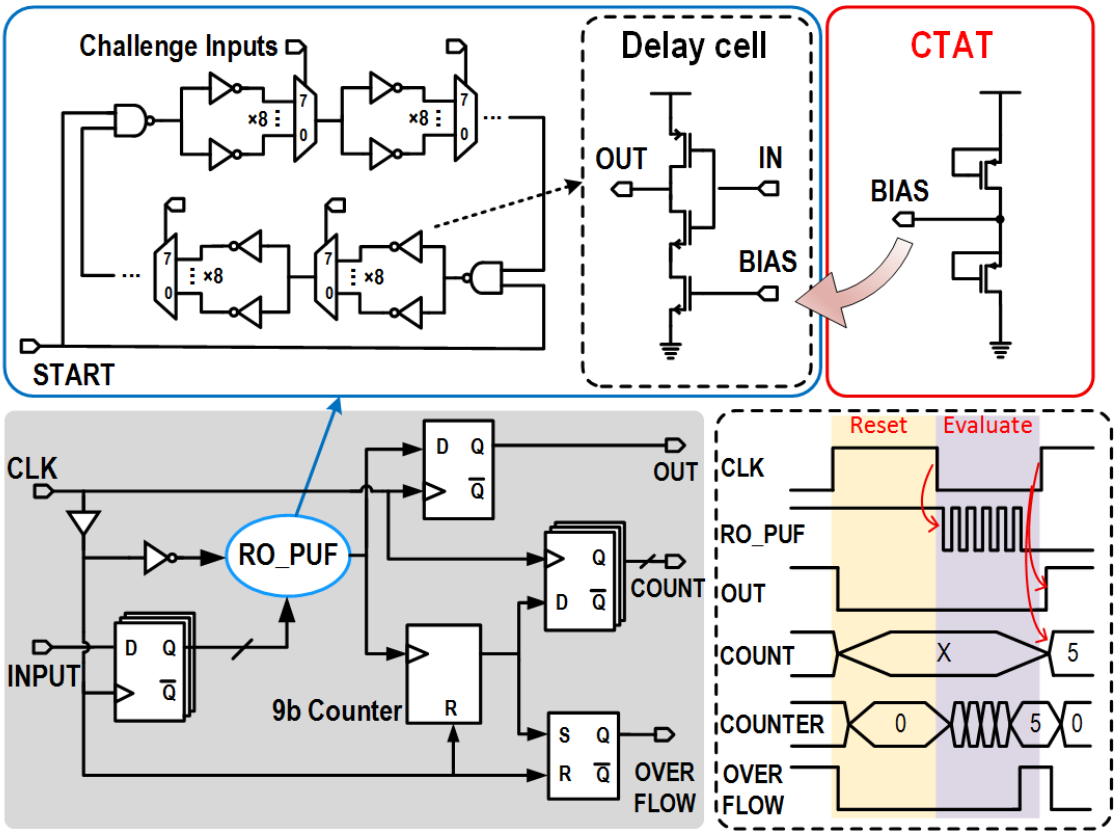


Figure 5.3: PUF block diagram with circuit implementations and operation waveforms.

since a slow collapse results from the two paths having nearly matched delay, making it more likely that the response is determined by noise, not process variation. Therefore, we implement a simple yet effective dynamic thresholding technique based on the number of cycles to collapse. A 9-bit counter in the PUF control logic (Figure 5.3) records the number of cycles to collapse. A counter bit is then used to determine if the count exceeds the set threshold (output OVERFLOW), in which case the PUF output is discarded.

Figure 5.4 shows the overall authentication protocol. During enrollment, a chip is interrogated; only CRPs with a collapse cycle smaller than the threshold are recorded. During verification, the chip is interrogated with a stored challenge. If the collapse count is larger than the threshold, the CRP is skipped and the next stored CRP is used. If the collapse count is smaller than threshold, the response is checked against the stored golden CRP. The process is repeated until either a response does not match and the authentication is rejected, or a sufficient number of CRPs match and the authentication is approved.

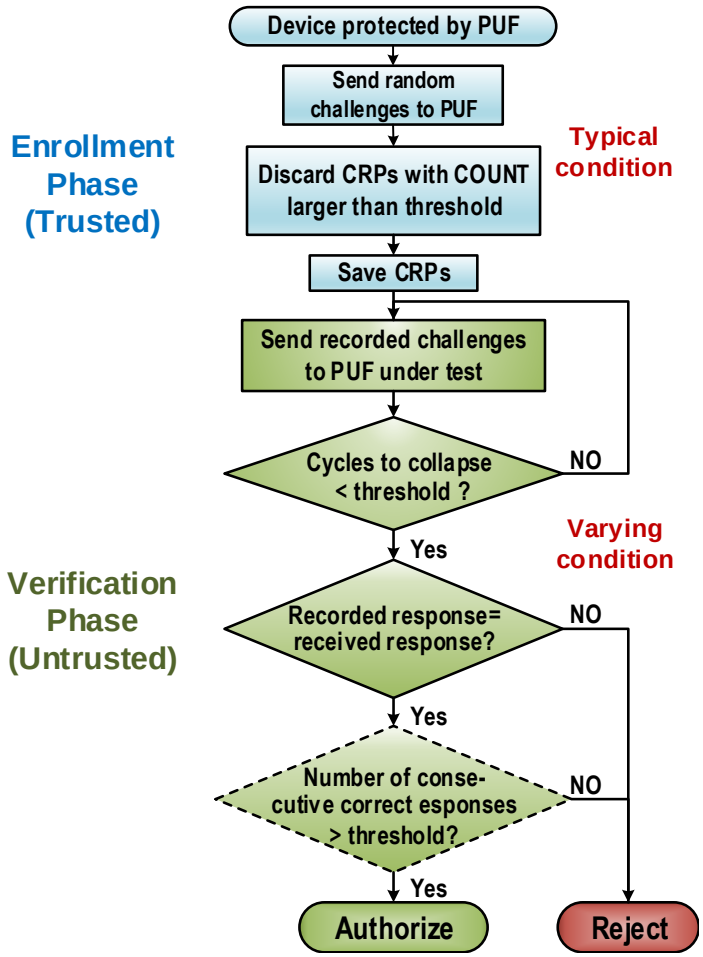


Figure 5.4: A basic PUF authentication protocol employing the dynamic thresholding technique.

5.3 Measurement Results

The proposed PUF is implemented in 40nm CMOS technology and a micrograph of the test chip is shown in Figure 5.5. Measurement results confirms the relationship between collapse time and bit error rate as illustrated in Figure 5.6. Thanks to the runtime indicator, unreliable PUF responses can be discarded in the runtime with dynamic thresholding. As shown in Figure 5.7, this technique dramatically reduces the bit-error rates (BER) in worst-case operating conditions (-25°C, 0.7V V_{DD}) from 9% to 0.002% with a threshold value of 32, or down to 0 with a threshold value of 16. Smaller thresholds provides lower BER at the cost of more discarded CRPs and more evaluations during enrollment and verification. However, even at a threshold of 16, the total discarded CRPs is an acceptable 34%, of which 80% is discarded during the initial enrollment phase. Also, PUF throughput is not necessarily reduced with a smaller threshold because the

global clock can run faster due to faster collapse. Note also that unlike other stabilization methods in [27,30], the proposed dynamic thresholding does not require any extra testing effort before each authentication.

Inter-chip and intra-chip Hamming distances (HD) are the other important metrics for a PUF, quantifying spatial uniqueness and temporal stability, respectively. Inter-chip HD are measured across 20 dies with 1000 challenges (Figure 5.8). Outputs are grouped into 100-bit keys; keys from different dies but the same challenge sets are compared in all possible pairs. Average normalized HD is 0.5007 ($\sigma = 0.0627$), very close to the ideal value of 0.5, which gives maximum uniqueness. Intra-chip Hamming distance is measured using 5000 challenges with each challenge evaluated 1000 times. The average intra-chip HD is 0.0101 ($\sigma = 0.0635$) at nominal conditions without dynamic thresholding. After applying dynamic thresholding with a threshold value of 16, the BER and intra-die HD remain 0. Even without thresholding, a 50$\times$ mean value separation between inter and intra-die HD is sufficient to authenticate a PUF with a failure probability as low as 1.2×10^{-30} (assume 109 chips, 256-bit response and a tolerance of 15 error bits) [23]. In this case, false alarm rate (FAR) and false detection rate (FDR) are 1.16×10^{-39} and 2×10^{-73}, respectively. Moreover, average HD between responses of different challenges on same die is 0.4722 ($\sigma = 0.0496$), showing good uniqueness among CRPs (Figure 5.8, upper right). Measurement results

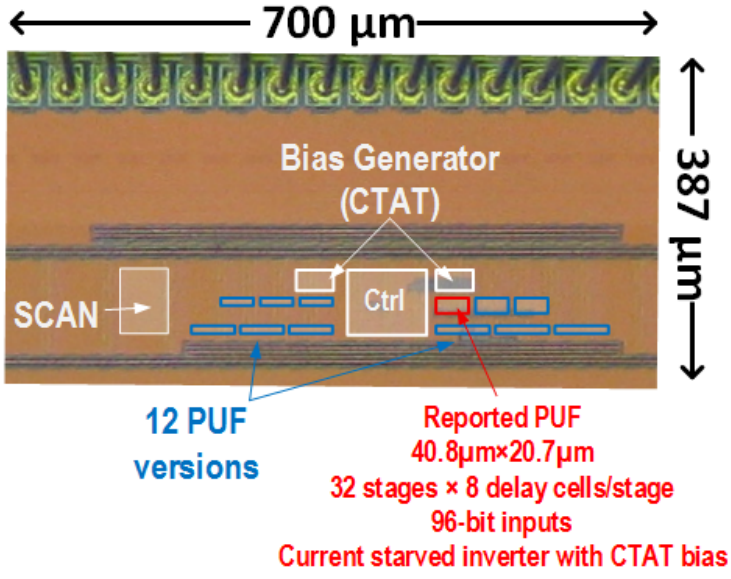


Figure 5.5: Die micrograph of 40nm CMOS PUF test chip.

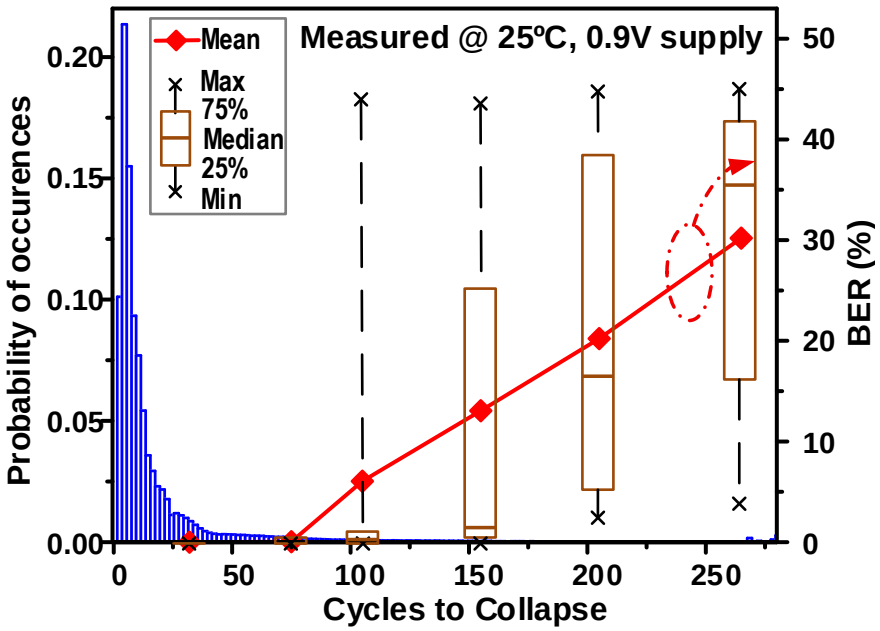


Figure 5.6: Measured distribution of cycles to collapse and its relation with bit error rates.

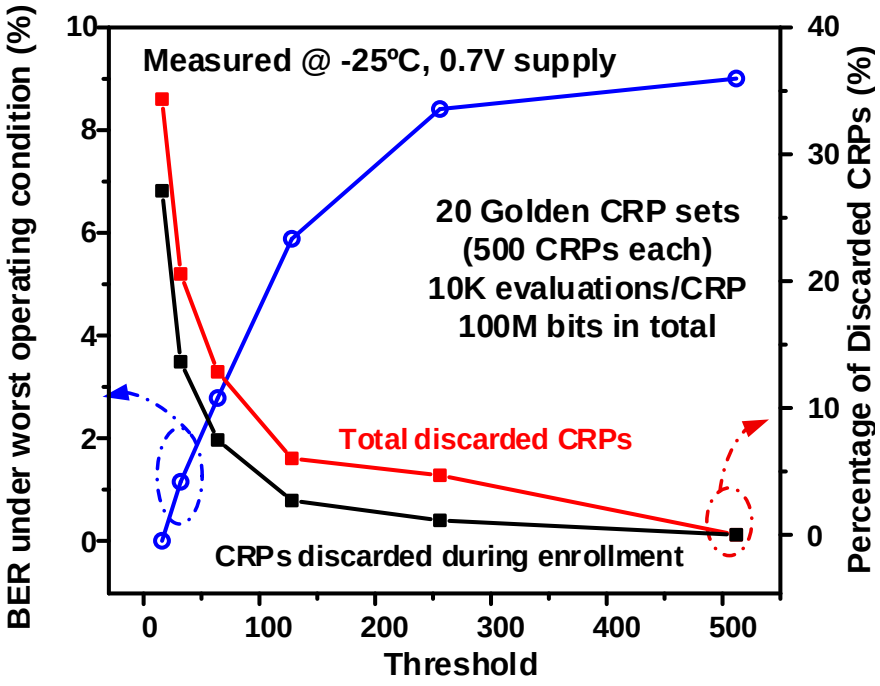


Figure 5.7: Measured distribution of cycles to collapse and its relation with bit error rates.

also show that autocorrelation of PUF responses is 0.0283 with 95% confidence (Figure 5.8, lower right).

A PUF will experience significant operating condition variations in a host device, especially

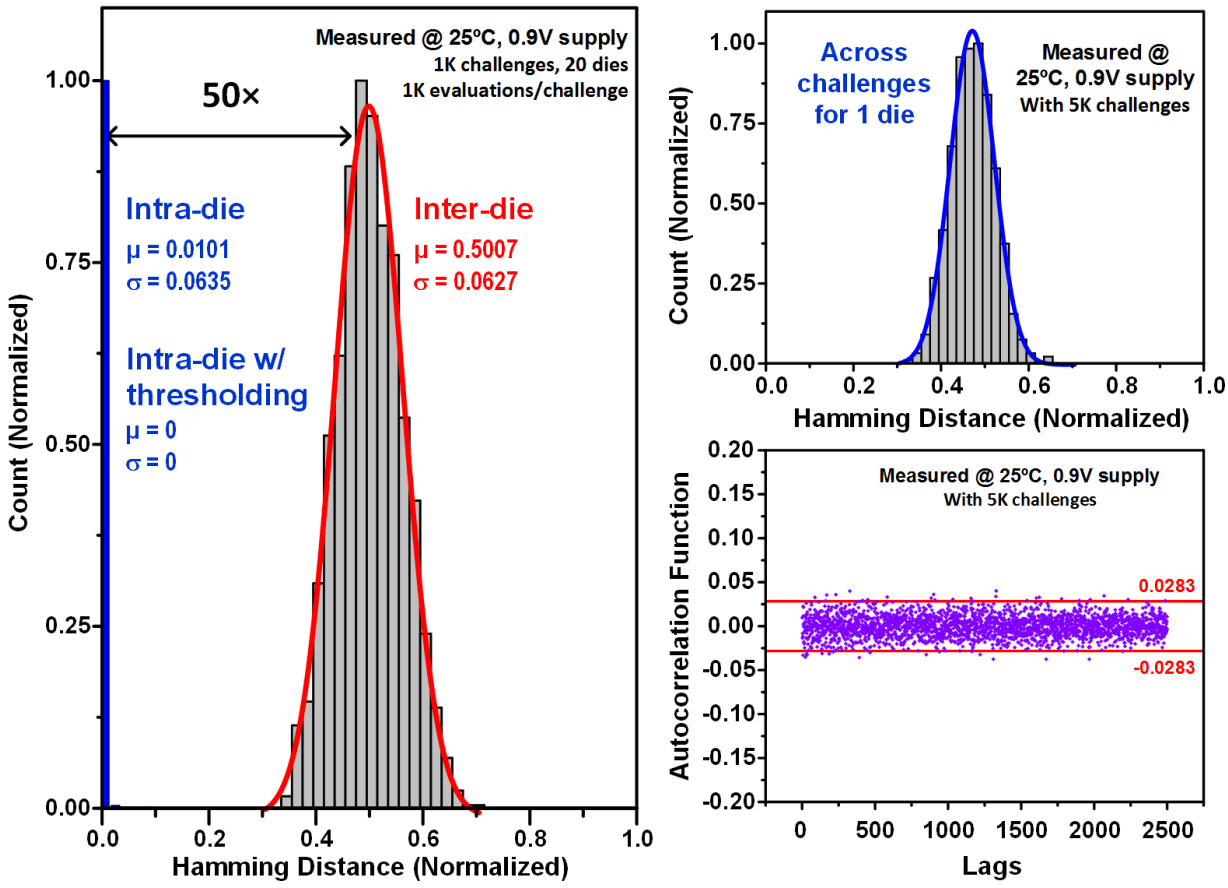


Figure 5.8: Measured intra-die and inter-die Hamming distances and autocorrelation function for responses to same challenges across 20 dies and responses to different challenges on a single chip.

for Internet-of-Things devices. For secure authentication, bit flipping due to varying environments should be minimized. Thanks to the current-starved delay cells, CTAT bias voltage, and dynamic thresholding, the proposed PUF maintains a $< 10^{-8}$ BER across 25 to 125$^{\circ}C$ and 0.7 to 1.2V ranges with the golden CRPs generated at nominal 25$^{\circ}C$ and 0.9V (Figure 5.9). The percentage of discarded CRPs at different combination of voltage and temperature conditions are plotted in Figure 5.9 as well.

In 40nm CMOS, the PUF generates response bits at $\sim 1.6Mb/s$ while consuming $28.4\mu W$. The design occupies only $845\mu m^2$ to provide around 5.5×10^{28} possible CRPs after threholding. Figure 5.1 summarizes the PUF measurement results and compares it to prior arts.

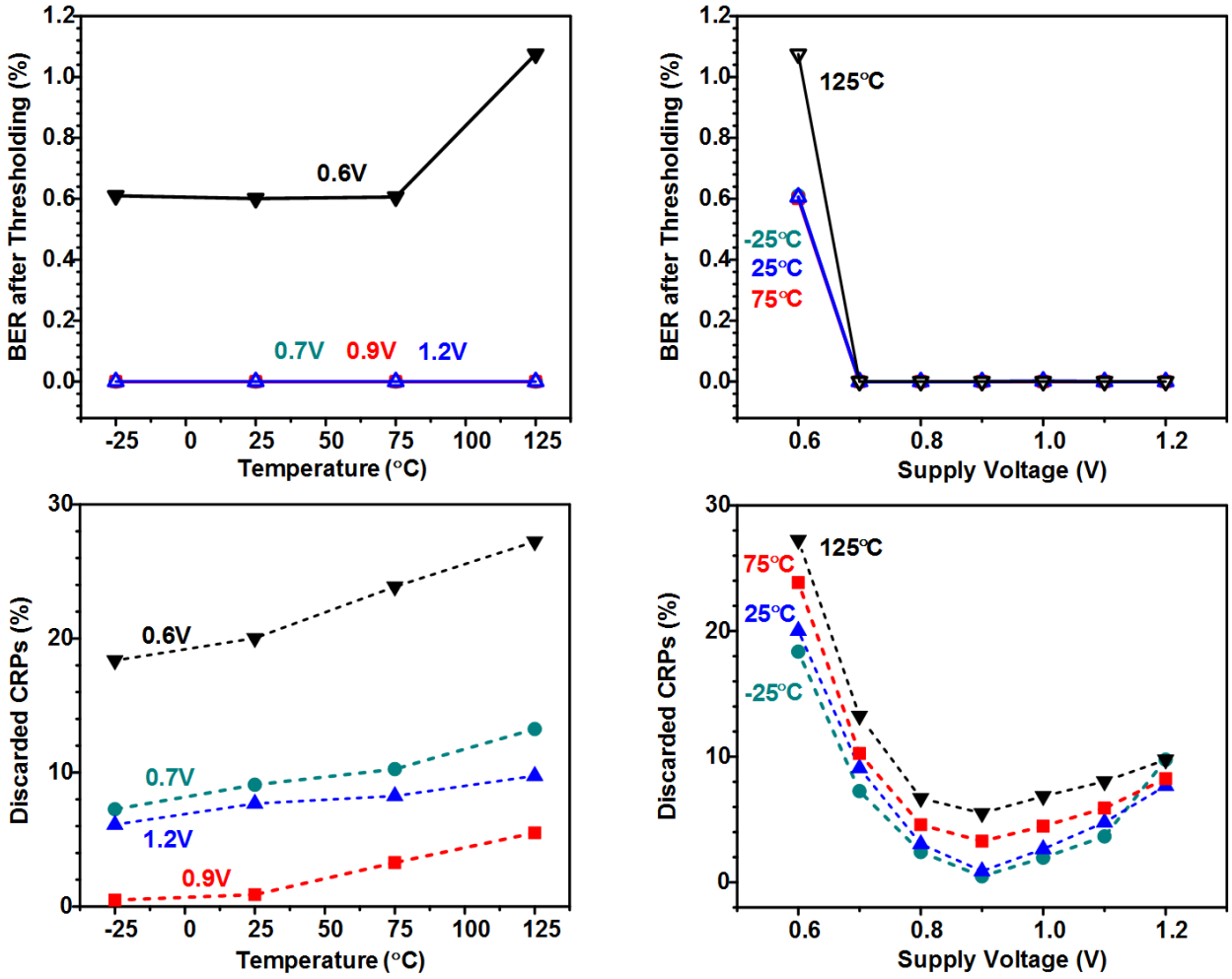


Figure 5.9: Measured BER and discarded CRPs over temperature and supply voltage variations, using threshold value of 16.

5.4 Summary

In summary, a "strong" PUF based on frequency collapse in even-stage ring oscillators is presented to improve the reproducibility of "strong" PUFs. The collapse time serves as a runtime indicator of the confidence level of the current PUF response and enables a dynamic thresholding technique that significantly improves the bit error rate of the PUF across wide temperature and voltage variations (25 to 125°C and 0.7 to 1.2V). The dynamic thresholding technique also reduces error correction computation and memory cost compared to conventional stabilizing techniques. The high reproducibility also significantly relieves one constraints in protocol designs of using PUFs for authentication and secret key storage.

Table 5.1: Summary of measurement results and a comparison with state-of-the-art silicon PUF and chip ID designs.

		This work (25°C, 0.9V core supply)	JSSC 11 [30]	VLSI 04 [23]	DAC 07 [29]	ISSCC 14 [27]	VLSI 10 [31]	JSSC 08 [28]
Technology		**40nm**	90nm	0.18μm	FPGA	22nm	65nm	0.13μm
Architecture		**Digital**	Analog	Digital	Synthesized	Digital	Digital	Digital
Number of possible CRPs		**~5.5×10^{28}**	10^{25}	1.4×10^{20}	523776	1 (Chip ID)	1(Chip ID)	1(Chip ID)
BER in Typical Case		**0**	0.009%	0.7%	-	-	0	3.04%
Tested Operating Conditions	**Temp(°C)**	**-25~125**	25~125	27~67	-20~120	25~50	0~85	-
	Supply	**0.7~1.2V**	±10%	±2%	±10%	0.7~0.9V	±10%	-
BER in Worst Case		**< 1×10^{-8} ***	0.1%	4.8%	0.48%	0.97%**	0	-
Bit Rate (Mb/s)		**1.6*****	0.00625	20	-	2000	625	1
Core Area (μm^2)		**845**	35000	-	-	4.66*256b= 1193	1242	15288
Power (μW)		**28.4**	38	-	-	25	212.5	0.137
Efficiency (pJ/bit)		**17.75**	6080	-	-	0.19	0.34	1.6
Stabilizing Method		**Dynamic thresholding**	Mask	Voting	-	Voting, Burn-in, Mask, ECC	-	-

* Threshold is 16 for this BER measurement and BER is 0 in 100M bits tested.
** After ECC with BCH code, BER is 0.
*** Effective throughput = Clock frequency × (1 - Percentage of CRPs discarded during evaluation in worst condition).

CHAPTER 6

Exploiting the Analog Properties of Digital Circuits for Malicious Hardware

6.1 Introduction

Hardware is the base of a computing system. All software executes on top of a processor. Software must trust hardware to faithfully implement the instructions. For many types of hardware flaws, software has no way to check if something went wrong [32, 33]. Even worse, if there is an attack in hardware, it can contaminate all layers of a system that depend on that hardware—violating high-level security policies correctly implemented by software.

The trend towards smaller transistors while beneficial for increased performance and lower power, has made fabricating a chip expensive. With every generation of transistor comes the cost of retooling for that smaller transistor. For example, it costs 15% more to setup the fabrication line for each successive process node and by 2020 it is expected that setting-up a fabrication line for the smallest transistor size will require a $20,000,000,000 upfront investment [34]. To amortize the cost of the initial tooling required to support a given transistor size, most hardware companies outsource fabrication.

Outsourcing of chip fabrication opens-up hardware to attack. The most pernicious fabrication-time attack is the dopant-level Trojan [35, 36]. Dopant-level Trojans convert trusted circuitry into malicious circuitry by changing the dopant ratio on the input pins to victim transistors. This effectively ties the input of the victim transistors to a logic level 0 or 1—a short circuit. Converting

existing circuits makes dopant-level Trojans very difficult to detect since there are no added or removed gates or wires. In fact, detecting dopant-level Trojans requires a complete chip delayering and comprehensive imaging with a scanning electron microscope [37]. Unfortunately, this elusiveness comes at the cost of expressiveness. Dopant-level Trojans are limited by existing circuits, making it difficult to implement sophisticated attack triggers [36]. The lack of a sophisticated trigger means that dopant-level Trojans are more detectable by post-fabrication functional testing. Thus, dopant-level Trojans represent an extreme on a tradeoff space between detectability during physical inspection and detectability during testing.

To defend against malicious hardware inserted during fabrication, researchers have proposed two fundamental defenses: 1) use side-channel information (e.g., power and temperature) to characterize acceptable behavior in an effort to detect anomalous (i.e., malicious) behavior [38–41] and 2) add sensors to the chip that measure and characterize features of the chip's behavior (e.g., signal propagation delay) in order to identify dramatic changes in those features (presumably caused by activation of a malicious circuit) [42–44]. Using side channels as a defense works well against large Trojans added to purely combinational circuits where it is possible to test all inputs and there exists a reference chip to compare against. While this accurately describes most existing fabrication-time attacks, we show that it is possible to implement a stealthy and powerful processor attack using only a single added gate. Adding sensors to the design would seem to adapt the side-channel approach to more complex, combinational circuits, but we design an attack that operates in the analog domain until it directly modifies processor state, without affecting features measured by existing on-chip sensors.

We create a novel fabrication-time attack that is controllable, stealthy, and small. To make our attack controllable and stealthy we borrow the idea of counter-based triggers commonly used to hide design-time malicious hardware [45, 46] and adapt it to fabrication-time. To make our attack small, we replace the hundreds of gates required by conventional counter-based triggers implemented using digital logic with analog components—a capacitor and a few transistors wrapped-up in a single gate. Our attack works by siphoning charge from a target wire every time it toggles and storing that charge in a capacitor. If the wire toggles infrequently, the capacitor voltage stays near zero volts due to natural charge leakage. When the wire toggles frequently, charge accumulates on the capacitor—faster than it leaks away, eventually fully charging the capacitor. When the voltage

on the capacitor rises above a threshold, it deploys the payload—whose output is attached to a flip-flop changing that victim flip-flop to any desired value.

To demonstrate that our attack works for real chips, we implement a privilege escalation attack in the OR1200 [47] open source processor. We attach our capacitor to a signal that infrequently toggles with normal software, but toggles at a high rate with specially-crafted, usermode trigger programs. For our victim flip-flop, we select the privilege bit (i.e., user or supervisor mode). Because the attack taps into both the digital layer and the analog layer, it is unable to be simulated completely using existing tools that operate at only a single layer. As such, we fabricate our malicious processor to verify its end-to-end operation. Experiments with our fabricated malicious processor show that it is trivial for a knowing attacker to activate the attack and escalate the privilege of their unprivileged process—all from usermode code, without operating system intervention. Experiments with an array of embedded systems benchmarks [48] show that it is unlikely that arbitrary software will trigger our attack.

This work presents three contributions:

1. We design and implement the first fabrication-time processor attack that mimics the triggered attacks often added during design time. As a part of our implementation, we are the first to show how a fabrication-time attacker can leverage the empty space common to Application-Specific Integrated Circuit (ASIC) layouts to implement malicious circuits.

2. We are the first to show how an analog attack can be much smaller and more stealthy than its digital counterpart. Our attack diverts charge from unlikely signal transitions to implement its trigger, thus, it is invisible to all known side-channel defenses. Additionally, as an analog circuit, our attack is below the digital layer and missed by functional verification performed on the hardware description language. Moreover, our attack relies on a complex and unlikely analog trigger sequence, thus, it is impractical to simulate at the analog level—which motivated us to fabricate a chip to verify that our attacks worked.

3. We fabricate the first openly malicious processor and then evaluate the behavior of our fabricated attacks across many chips and changes in environmental conditions. We compare these results to SPICE simulation models [1].

[1] We make both the software and hardware code pertaining to A2 publicly available [49].

6.2 Background

The focus of this paper is fabrication-time attacks that leverage analog characteristics of integrated circuits as a trigger. In this section, we start with an overview of the integrated circuit (IC) design process and possible malicious attacks at different phases. Then we discuss the threat model of our proposed attack.

6.2.1 Integrated Circuit Design Process

Figure 6.1 shows the typical design process of integrated circuits [50]. This process often involves collaboration between different parties all over the world and each step is likely done by different teams even if they are in same company, which makes it vulnerable to malicious attacks by rogue engineers involved in any of the above steps.

6.2.2 Threat Model

It is possible to implement our attack at either the back-end phase or at the fabrication phase. Since it is strictly more challenging to implement attacks at the fabrication phase due to limited information and ability to modify the design compared to the back-end phase, we focus on that threat model.

The attacker starts with a Graphic Database System II (GDSII) file that is a polygon representation of the completely laid-out and routed circuit. This is a very restrictive threat model as it means that the attacker can only modify existing circuits or—as we are the first to show in this paper—add attack circuits to open spaces in the laid-out design. The attacker can *not* increase the dimensions of the chip or move existing components around. This restrictive threat model also means that the attacker must perform some reverse engineering to select viable victim flip-flops and wires to tap. As detailed in Section 6.6.3, a public specification of the chip to be fabricated makes this process easier. After the untrusted fabrication house completes fabrication, it sends the fabricated chips off to a trusted party for post-fabrication testing. Our threat model assumes that the attacker has no knowledge of the test cases used for post-fabrication testing, which dictates the use of a sophisticated trigger to hide the attack.

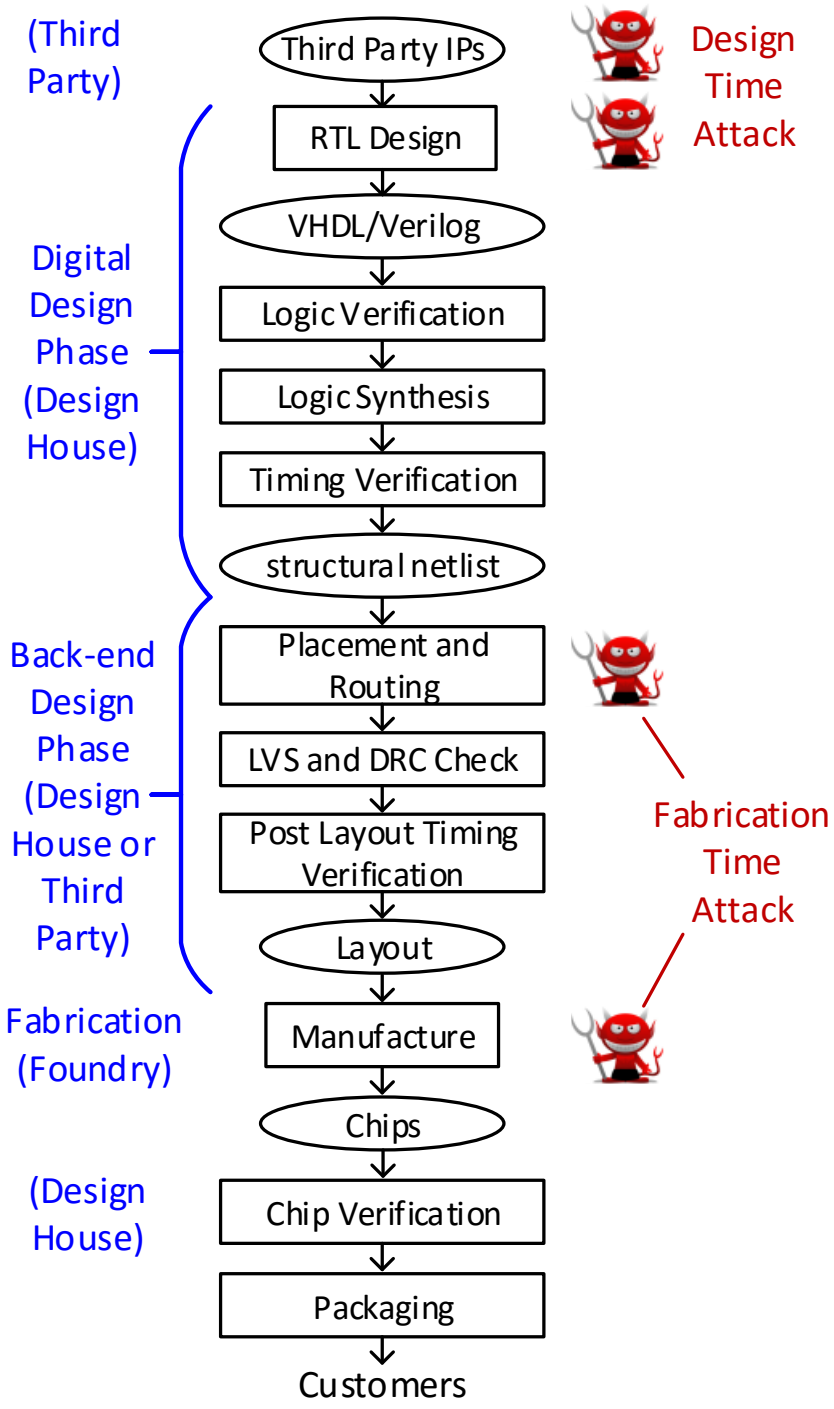


Figure 6.1: Typical IC design process with commonly-research threat vectors highlighted in red. The blue text and brackets highlights the party in control of the stage(s).

Leading up to the attacker getting a GDSII file, our threat model assumes that a design house correctly implements the specification for the chip's behavior in some hardware description language (HDL). Once the specification is implemented in an HDL and that implementation has been verified, the design is passed to a back-end house. Our threat model assumes that the back-end house—who places and routes the circuit—is also trusted. This means that the delivered GDSII file represents a perfect implementation—at the digital level of abstraction—of the chip's specification. The attacker is free to modify the design at both the digital level by adding, removing, or altering circuits and at the analog level (e.g., increasing electromagnetic coupling of wires through layout or adding analog components).

Note that the chip vendor is free to run any additional tests on the fabricated chip. We assume that the attacker has no knowledge or control about post-fabrication testing. We only assume that testing is bound by the limits of practicality.

6.3 Attack Methods

A hardware attack is composed of a trigger and a payload. The trigger monitors wires and state within the design and activates the attack payload under very rare conditions such that the attack stays hidden during normal operation and testing. Previous research has identified that evading detection is a a critical property for hardware Trojans designers [51]. Evading detection involves more than just avoiding attack activation during normal operation and testing though, it includes hiding from visual/side-channel inspection. There is a tradeoff at play between the two in that the more complex the trigger (i.e., the better that it hides at run time), the larger the impact that trigger has on the surrounding circuit (i.e., the worse that it hides from visual/side-channel inspection).

We propose A2, a fabrication-time attack that is small, stealthy, and controllable. To achieve these outcomes, we develop trigger circuits that operate in the analog domain; circuits based on charge accumulating on a capacitor from infrequent events inside the processor. If the charge-coupled infrequent events occur frequently enough, the capacitor will fully charge and the payload is activated, which deploys a privilege escalation attack. We target the privilege bit in a processor, as privilege escalation constitutes a simple payload with maximum capability provided to the attacker. Our analog trigger similar to the counter-based triggers often used in digital triggers, except

using the capacitor has the advantage of a natural reset condition due to leakage.

We create the trigger using a custom analog circuit that a fabrication-time attacker inserts after the entire design has been placed and routed. Compared to traditional digitally described hardware Trojans, our analog trigger maintains a high level of stealth and controllability, while dramatically reducing the impact on area, power, and timing due to the attack. An added benefit of a fabrication-time attack compared to a design time attack (when digital-only triggers tend to get added) is that the fabrication-time attack has to pass through few verification stages.

To highlight the design space of our analog trigger technique, we show how an attacker can connect several simple trigger circuits to create arbitrary trigger patterns to improve secrecy and/or controllability. In addition to the number of stages, we show how an attacker can tune several design parameters to achieve trade-offs between the ease of triggering the payload and its stealthiness, even to the point of creating triggers that can only be expressed under certain process variation and/or environmental conditions. This trade-off space is only possible through the use of an analog-based trigger.

In the following sections, we describe the design and working principles of our analog trigger. We present the designs of both a base single-stage trigger and a more complex, but flexible, multi-stage trigger. We also describe our privilege escalation attack which also has analog components. We conclude with an analysis of how an attacker, bounded by our threat model, would go about attacking a processor.

6.3.1 Single Stage Trigger Circuit

Based on our threat model, the high-level design objectives of our analog trigger circuit are as follows:

1. Functionality: The trigger circuit must be able to detect toggling events of a target victim wire similar to a digital counter and the trigger circuit should be able to reset itself if the trigger sequence is not completed in a timely manner.

2. Small area: The trigger circuit should be small enough to be inserted into the empty space of an arbitrary chip layout after placement and routing of the entire design. Small area overhead also implies better chance to escape detection.

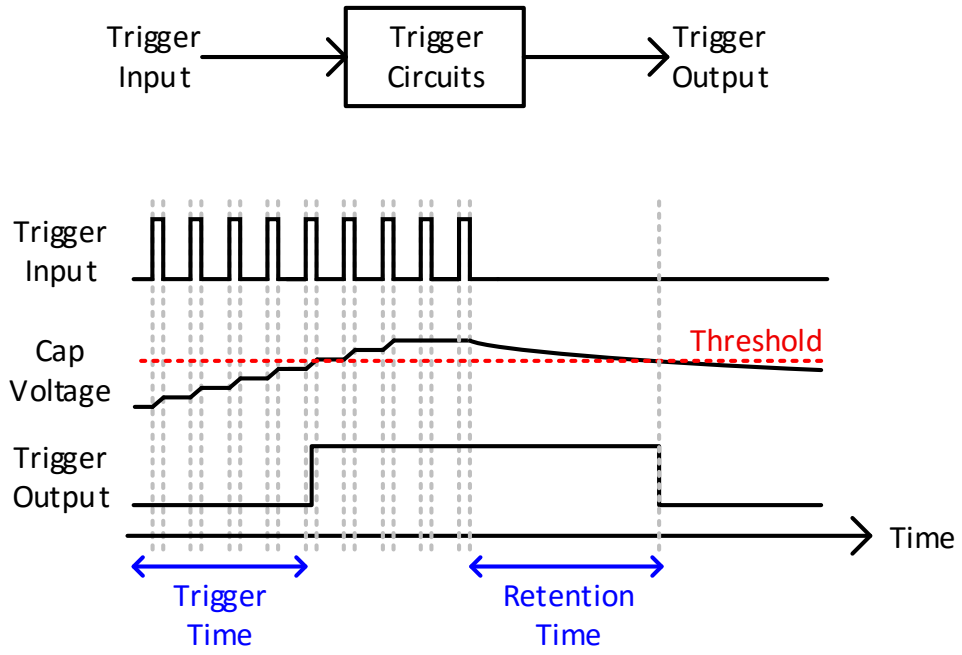


Figure 6.2: Behavior model of proposed analog trigger circuit.

3. Low power: The trigger circuit is always actively monitoring its target signals, therefore power consumption of the design must be minimized to hide it within the normal fluctuations of entire chip's power consumption.

4. Negligible timing perturbation: The added trigger circuit must not affect the timing constraints for common case operation and timing perturbations should not be easily separable from the noise common to path delays.

5. Standard cell compatibility: Since all digital designs are based on standard cells with fixed cell height, our analog trigger circuit should be able to fit into the standard cell height. In addition, typical standard cells use only metal layer 1 [2] for routing while higher metal layers are reserved for connections between cells, therefore it is desirable for the trigger circuit to use only metal layer 1 for easier insertion into final layout and detection more difficult.

To achieve these design objectives, we propose an attack based on charge accumulation inside capacitors. A capacitor acts as a counter which performs analog integration of charge from a victim

[2]Several layers of metal wires are used in modern CMOS technologies to connect cells together, lower level metal wires are closer to transistors at bottom and have smaller dimensions for dense but short interconnections, while higher metal layers are used for global routing. The lowest layer of metal is usually assigned as metal layer 1 and higher metal layers have correspondingly larger numbers.

wire while at the same time being able to reset itself through natural leakage of charge. A behavior model of charge accumulation based trigger circuits comprises 2 parts.

1. Charge accumulation: Every time the victim wire that feeds the trigger circuit's capacitor toggles (i.e., changes value), the capacitor increases in voltage by some ΔV. After a number of toggles, the capacitor's voltage exceeds a predefined threshold voltage and enables the trigger's output—deploying the attack payload. The time it takes to activate fully the trigger is defined as *trigger time* as shown in Figure 6.2. *Trigger time* equals toggling frequency of input victim wire multiplied by the number of consecutive toggles to fill the capacitor.

2. Charge leakage: A leakage current exists over all time that dumps charge from the trigger circuit's capacitor, reducing the capacitor's voltage. The attacker systematically designs the capacitor's leakage and accumulation such that leakage is weaker than charge accumulation, but just enough to meet some desired *trigger time*. When the trigger input is inactive, leakage gradually reduces the capacitor's voltage even eventually disabling an already activated trigger. This mechanism ensures that the attack is not expressed when no intentional attack happens. The time it takes to reset trigger output after trigger input stops toggling is defined as *retention time* as shown in Figure 6.2.

Because of leakage, a minimum toggling frequency must be reached to successfully trigger the attack. At minimum toggling frequency, charge added in each cycle equals charge leaked away. *trigger time* is dependent on toggling frequency, lower toggling rate requires more cycles to trigger because more charge is leaked away each cycle, meaning less charge accumulated on the capacitor each cycle. *retention time* is only dependent on the strength of leakage current. *Trigger time* and *retention time* are the two main design parameters in our proposed analog trigger attack circuits that we can make use of to create flexible trigger conditions and more complicated trigger pattern as discussed in Section 6.3.2. A stricter triggering condition (i.e., faster toggling rate and more toggling cycles) reduces the probability of a false trigger during normal operation or post-fabrication testing, but non-idealities in circuits and process, temperature and voltage variations (PVT variations) can cause the attack to fail—impossible to trigger or trivial to accidentally trigger—for some chips. As a result, a trade-off should be made here between a reliable attack that can be expressed in every manufactured chip under varying environmental conditions and a more stealthy attack

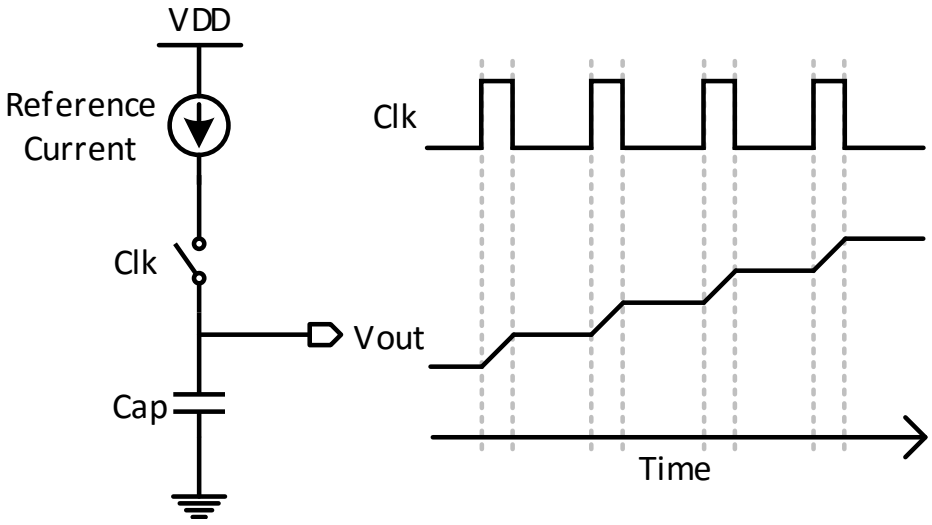


Figure 6.3: Concepts of conventional charge pump design and waveform.

that can only be triggered for certain chips, under certain environmental conditions, and/or very fast toggling rate of trigger inputs generated by software.

To find a physical implementation of the function described previously, we first try a charge pump widely used in phase locked loop (PLL) designs as shown in Figure 6.3. *Clk* in the figure represents some toggling wire that adds charge to *Cap* capacitor during positive phase of *Clk*. The voltage step added to *Cap* during one positive phase can be calculated as,

$$\Delta V = \frac{I_{ref} \times T_{positive}}{Cap} \tag{6.1}$$

This implies that the voltage on the cap can exceed a threshold in $V_{threshold}/\Delta V$ cycles. Due to our area and power requirements, we need to minimize I_{ref} and Cap size while maintaining an acceptable number of cycles to trigger the attack. There are 3 common methods to implement capacitors in CMOS technology: transistor gate oxide cap (MOS cap), metal-insulator-metal cap (MIM cap) and metal-oxide-metal (MOM cap). The other 2 options require higher metal layers and have less capacitance density, therefore we select the MOS cap option. Given the area constraints, our MOS cap can be at most tens of fF, which means the current reference should be in nA range. Such a current reference is nontrivial to design and varies greatly across process, temperature, and voltage variations. Therefore, we need a new circuit design to solve these problems for a more reliable and stealthy attack. However, the circuit in Figure 6.3 is useful for attacks that wish

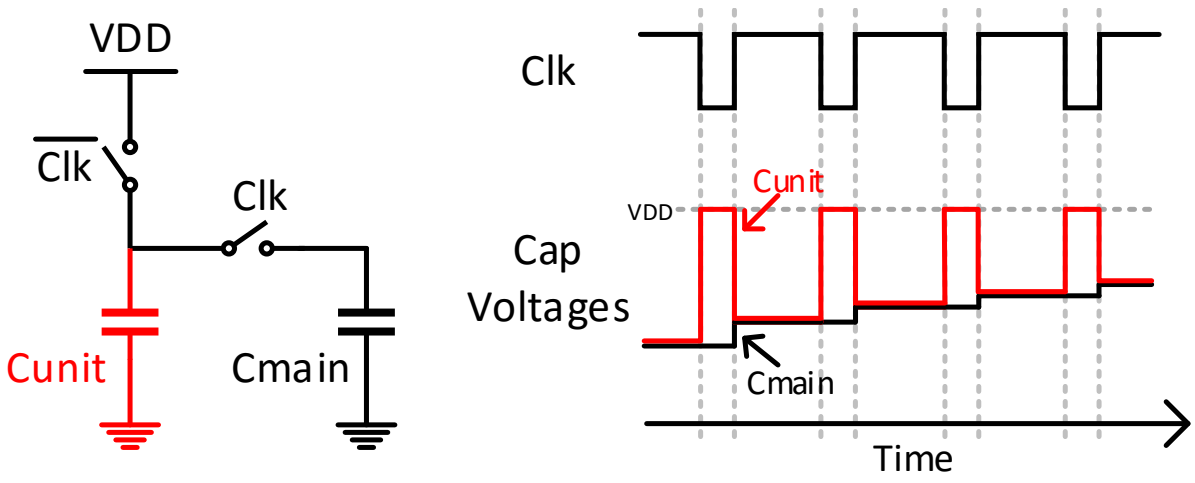


Figure 6.4: Design concepts of analog trigger circuit based on capacitor charge sharing.

to impact only a subset of manufactured chips or for scenarios where the attacker can cause the victim wire to toggle at a high rate for hundreds of cycles.

A new charge pump circuit specifically designed for the attack purpose is shown in Figure 6.4. Instead of using reference current and positive phase period of *Clk* to control ΔV, the new scheme uses one additional small unit capacitor *Cunit* to better control the amount of charge dumped on main capacitor each time. During the negative phase of *Clk*, *Cunit* is charged to VDD. Then during positive phase of *Clk*, the two capacitors are shorted together, causing the two capacitors to share charges. After charge sharing, final voltage of the two capacitors is the same and ΔV on *Cmain* is as,

$$\Delta V = \frac{Cunit \times (VDD - V_0))}{Cunit + Cmain} \tag{6.2}$$

where V_0 is initial voltage on *Cmain* before the transition happens. As can be seen, ΔV is decreasing as the voltage ramps up and the step size solely depends on the ratio of the capacitance of the two capacitors. We can achieve different *trigger time* values by sizing the two capacitors. Compared to the design in Figure 6.3, the new scheme is edge triggered rather than level triggered so that there is no requirement on the duty cycle of trigger inputs, making it more universal. The capacitor keeps leaking over time and finally ΔV equals the voltage drop due to leakage, which sets the maximum capacitor voltage.

A transistor level schematic of the proposed analog trigger circuit is shown in Figure 6.5. *Cunit*

and *Cmain* are implemented with MOS caps. $M0$ and $M1$ are the 2 switches in Figure 6.4. A detector is used to compare cap voltage with a threshold voltage and can be implemented in two simple ways as shown in Figure 6.6. One option is an inverter, which has a switching voltage depending on sizing of the two transistors and when the capacitor voltage is higher than the switching voltage, the output is 0; otherwise, the output is 1. The other option is a Schmitt trigger, which is a simple comparator with hysteresis. It has a large threshold when input goes from low to high and a small threshold when input goes from high to low. The hysteresis is beneficial for our attack, because it extends both *trigger time* and *retention time*.

In practice, all transistors have leakage currents even in their off state and our capacitors are very small, therefore the cap voltage is affected by leakage currents as well. To balance the leakage current through $M0$ and $M1$, an additional leakage path to ground (NMOS $M2$ in Figure 6.5) is added to the design. An attacker must carefully calculate all leakage paths flowing to and out of the capacitor node in order to balance their effects to achieve the *trigger time* and *retention time* targets. There are three major leakage paths in our analog trigger design: sub-threshold leakage current through switch $M1$, transistor $M2$, and gate tunneling leakage current (as shown in Figure 6.5). Because leakage currents are sensitive to process, voltage and temperature variations, balancing all the leakage paths is the most challenging part in the implementation of a reliable trigger analog trigger.

For the trigger circuit to work, capacitor voltage without any toggling on its input wire should be low enough to not, in any manufacturing or environmental corner case, be self-triggering. Also, the maximum voltage under the fastest rate of toggling by the victim wire that the attacker can produce must be enough to have a good margin for successful attack, allowing a wider range of acceptable toggling rates that will eventually trigger the attack. These conditions should be met under all PVT variations for a reliable attack, or under certain conditions if attacker only want the attack to be successful under these conditions. No matter what the design target is, minimum voltage should always be kept lower than threshold voltage to avoid exposing the attack in normal use.

A SPICE simulation waveform is shown in Figure 6.7 to illustrate the desired operation of our analog trigger circuit after optimization. The operation is same as the behavioral model that we proposed in Figure 6.2, indicating that we can use the behavior model for system-level attack

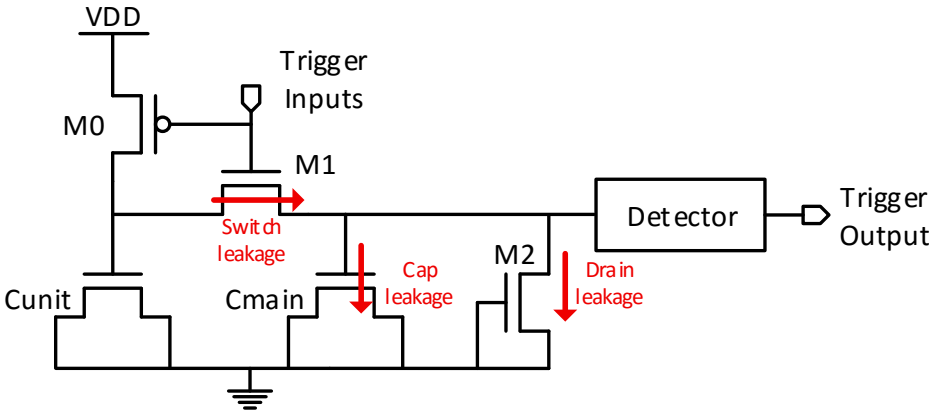


Figure 6.5: Transistor level schematic of analog trigger circuit.

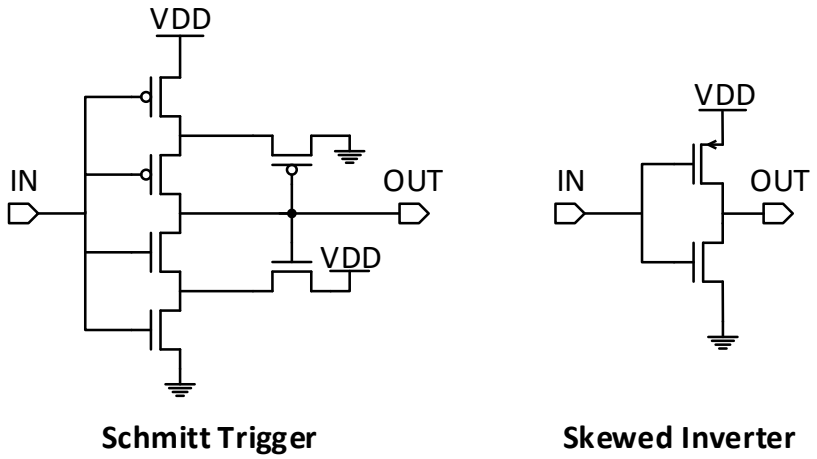


Figure 6.6: Schematics of detector circuits.

design.

6.3.2 Multi-stage Trigger Circuit

The one-stage trigger circuit described in the previous section takes only one victim wire as an input. Using only one trigger input limits the attacker in two ways:

1. False trigger activations: Because fast toggling of one signal for tens of cycles triggers the single stage attack, there is still a chance that normal operations or certain benchmarks can expose the attack. We can imagine cases where there is only a moderately controllable wire available. A single-stage trigger might be prone to false activations in such a scenario, but a multi-stage trigger could use wires that normally have mutually-exclusive toggle rates as

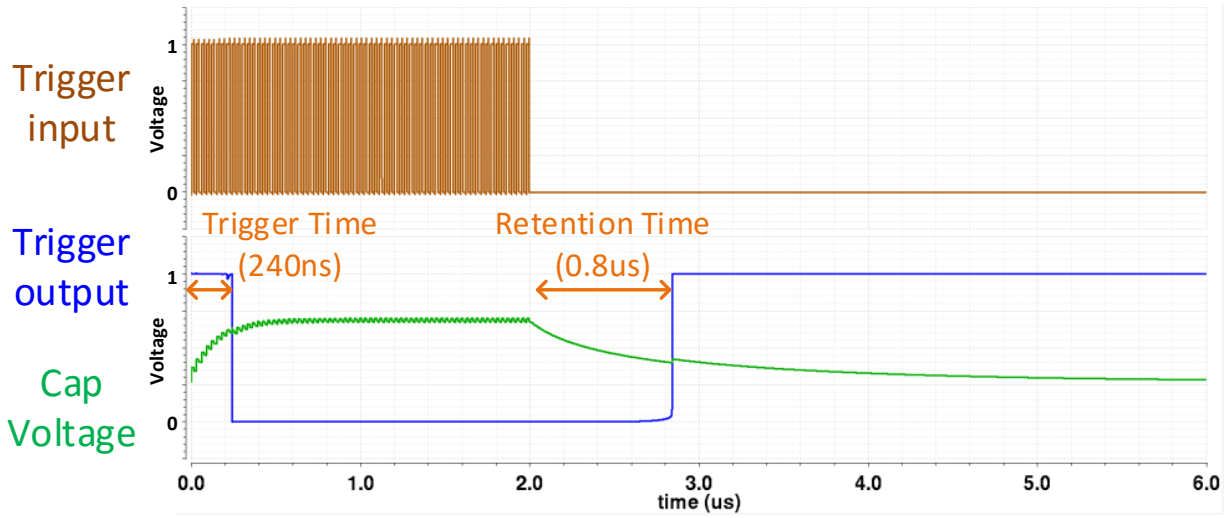


Figure 6.7: SPICE simulation waveform of analog trigger circuit.

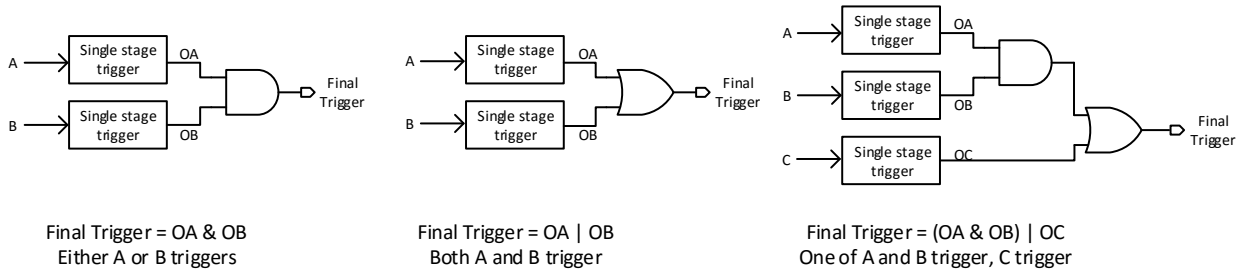


Figure 6.8: Basic ways of connecting single-stage triggers to form a multi-stage trigger.

inputs, making it stealthy and controllable.

2. Software flexibility: Certain instructions are required to cause fast toggling of the trigger input and there is not much room for flexible and stealthy implementation of the attack program. For example, some types of multi-stage triggers could support a wide range of attack programs. This would allow the attacker to repeatedly compromise a victim system.

To make the attack even more stealthy, we note that an attacker can make a logical combination of two or more single-stage trigger outputs to create a variety of more flexible multi-stage analog trigger. Basic operations to combine two triggers together are shown in Figure 6.8. When analyzing the behavior of logic operations on single stage trigger output, it should be noted that the single-stage trigger outputs 0 when trigger condition is met. Thus, for *AND* operation, the final trigger is activated when either *A* or *B* triggers fire. For *OR* operation, the final trigger is activated when both *A* and *B* triggers fire. It is possible for an attacker to combine these simple *AND*

and *OR*-connected single-stages triggers into an arbitrarily complex multi-level multi-stage trigger. Figure 6.8 show what such a trigger could look like, creating a two level multi-stage trigger with the logical expression $(OA\&OB)|OC$. This third trigger fires when trigger C and one of triggers A or B fire. Lastly, it is important to note that not only can the inputs A, B, and C be different, but the internal circuit parameters for each single-stage trigger can also be different (even though we treat them as identical for simplicity).

Due to the analog characteristics of the trigger circuits, timing constraints limit the construction of multi-stage triggers, but also make accidental trigger probability vanishingly rare. A single-stage trigger circuit has two timing parameters, *trigger time* and *retention time*. For *AND* operation, the timing constraint is same as for a single-stage trigger, because only one of the triggers must activate. For *OR* operation, there are two methods to trigger the attack: 1) alternatively run the instructions to toggle victim wires A and B or 2) run the instructions to toggle A first for enough cycles to activate the trigger and then run the instructions to trigger B. For the first method, the timing constraint is minimum toggling frequency, because adding n stages reduces the toggling frequency for each trigger circuit by n times. For the second method, the timing constraint is that *retention time* of the stage n should be larger than the total *trigger time* of the following stages stages.

6.3.3 Triggering the Attack

Once the trigger circuit is activated, payload circuits activate hidden state machines or overwrite digital values directly to cause failure or assist system-level attacks. The payload can also be extra delay or power consumption of target wires to leak information or cause failure. For A2, the payload design is independent of the trigger mechanism, so our proposed analog trigger is suitable for various payload designs to achieve different attacks. Since the goal of this work is to achieve a Trojan that is nearly invisible while providing a powerful foothold for a software-level attacker, we couple our analog triggers to a privilege escalation attack [52]. We propose a simple design to overwrite security critical registers directly as shown in Figure 6.9. In any practical chip design, registers have asynchronous set or/and reset pins for system reset. These reset signals are asynchronous with no timing constraints so that adding one gate into the reset signal of one register

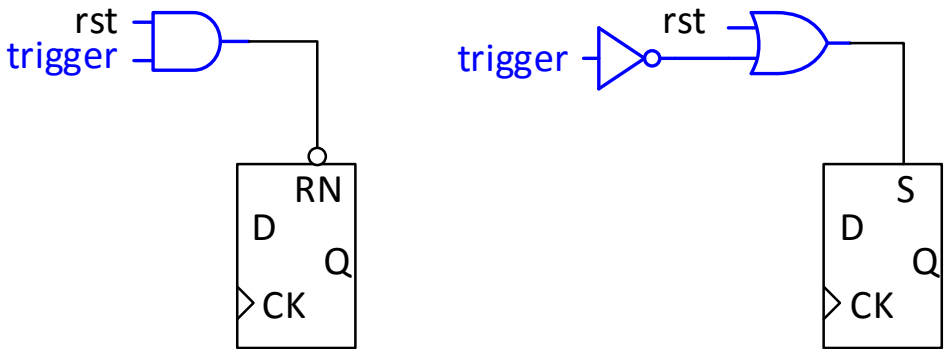


Figure 6.9: Design of payload to overwrite register value. Gates in blue lines are inserted for attack.

does not affect functionality or timing constraints of the design. Since the analog trigger circuit output is 0 when activated, we insert an *AND* gate between the existing reset wire and our victim flip-flop for active-low reset flops and we insert a *NOR* gate for for active-high set flops. Moreover, because there are no timing constraints on asynchronous inputs, the payload circuit can be inserted manually after final placement and routing together with the analog trigger circuits in a manner consistent with our threat model.

6.3.4 Selecting Victims

It is important that the attacker validate their choice of victim signal; this requires showing that the victim wire has low baseline activity and its activity level is controllable given the expected level of access of the attacker. To validate that the victim wire used in A2 has a low background activity, we use benchmarks from the MiBench embedded systems benchmark suite. We select these benchmarks due to their diverse set of workload characteristics and because they run on our resource-limited implementation. We expect that in a real-world attack scenario, the attacker will validate using software and inputs that are representative of the common case given the end use for the attacked processor. For cases where the attacker does not have access to such software or the attacked processor will see a wide range of use, the attacker can follow A2's example and use a multi-stage trigger with wires that toggle in a mutually-exclusive fashion and require inputs that are unlikely to be produced using off-the-shelf tools (e.g., GCC).

Validating that the victim wire is controllable requires that the attacker reason about their expected level of access to the end user system for the attacked processor. In A2, we assume that the

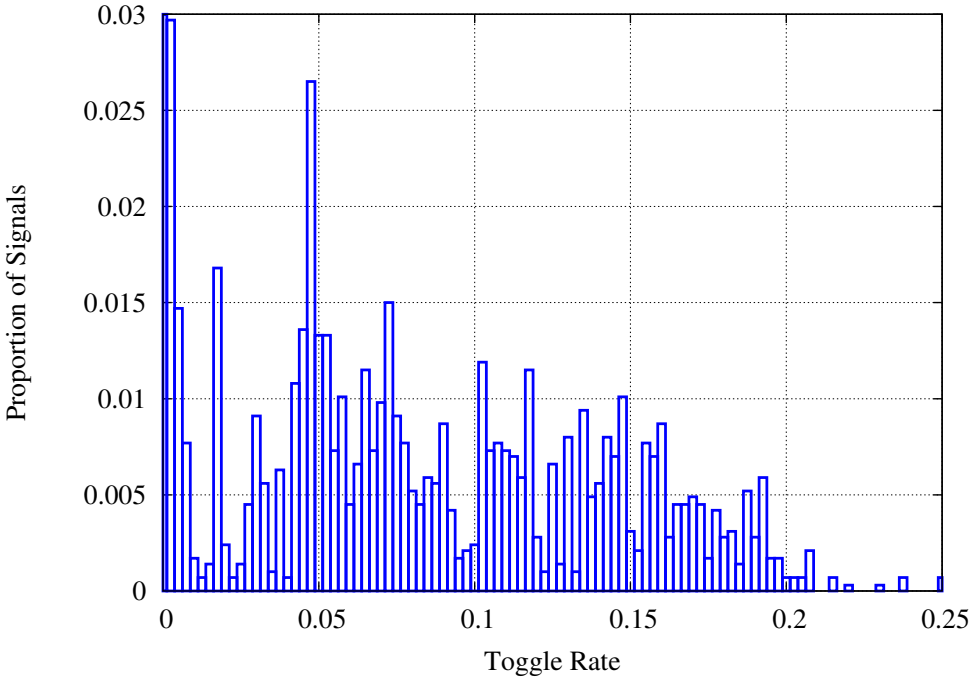


Figure 6.10: Distribution of paths toggling rate when running a benchmark program.

attacker can load and execute any unprivileged instruction. This allows us to create hand-crafted assembly sequences that activate the attack—remember that we selected victim wires that off-the-shelf tools will not produce code significantly activates. While this model works for attackers that have an account on the system, attackers in a virtual machine, or even attackers that can convince users to load code, we did not explore the challenges of less controllable attack scenarios. Two examples are triggering the attack from Javascript and triggering the attack situationally (e.g., radar containing the attacked chip senses a certain type of plane). We expect that our attack supports such triggering scenarios as there is no fundamental difference from running handcrafted unprivileged code: executable code contains a multitude of different instructions and different instructions activate different sets of wires in the processor. The difference is just an extra layer of abstraction. One challenge that we anticipate is the extra layer of abstraction will likely reduce the range of activity on potential victim wires. Our experimental results show that the attacker can deal with this by changing parameters of the analog trigger or even through careful use of a multi-stage trigger.

6.4 Implementation

To experimentally verify A2, we implement and fabricate it inside an open source processor with the proposed analog Trojans inserted in 65nm General Purpose (GP) CMOS technology. Because of the time and monetary cost of hardware fabrication, we include multiple attacks in each chip. One set of attacks are Trojans aimed at exposing A2's end-to-end operation, while the other set of attacks are implemented outside the processor, directly connected to IO pins so that we can investigate trigger behavior directly. In this section, we detail the selection of the trigger and attack payload in an OR1200 processor, the activity trigger insertion flow, and analog trigger testing structures.

6.4.1 Attacking a Real Processor

We implemented a complete open source OR1200 processor [47] to verify our complete attack including software triggers, analog triggers and payload. The OR1200 CPU is an implementation of the 32-bit OR1K instruction set with 5-stage pipeline. The implemented system in silicon consists of OR1200 core with 128B instruction cache and an embedded 128KB main program memory connected through a Wishbone bus. Standard JTAG interface and custom scan chain are implemented to load program, control and monitor the processor.

The OR1K instruction set specifies the existence of a privileged register called the Supervision Register (SR). The SR contains bits that control how the processor operates (e.g., MMUs and caches enabled) and flags (e.g., carry flag). One particular bit is interesting for security purposes; SR[0] controls the privilege mode of user, with 0 denoting user mode and 1 denoting supervisor mode. By overwriting the value of this register, an attacker can escalate a usermode process to supervisor mode as a backdoor to deploy various high-level attacks [51, 52]. Therefore, we make the payload of our attack setting this bit in the SR to 1 to give a usermode process full control over the processor. In order to evaluate both the one-stage and two-stage triggers described earlier, we have our two-stage triggered attack target SR[1]. Normally, this register bit controls whether timer-tick exceptions are enabled, but since our system does not use the timer and SR[1] requires privileged software to change its value, it is a simple way to know if our two-stage attack works.

Our analog trigger circuits require trigger inputs that can have a high switching activity under

certain (attacker) programs, but are almost inactive during testing or common case operation so that the Trojan is not exposed [3]. To search for suitable victim wires to serve as trigger inputs, we run a series of programs on the target processor in a HDL simulator, capturing the toggling rates of all wire. Figure 6.10 shows a histogram of wire toggling rates for the basicmath benchmark from MiBench (see Section 6.5). As the figure shows, approximately 3% of total wires in the OR1200 have nearly zero activity rate, which provides a wide range of options for an attacker. The target signals must also be easy to control by attack programs. To find the low activity wires for attacker controllability, we simulate our attack program in the same setup and identify the wires whose toggle rates increased dramatically. In our attack, we select divide by zero flag signal as the trigger for one-stage attack, because it is unlikely for normal programs to continuously perform division-by-zero while it is simple for an attacker to deliberately perform such operations in a tight loop. Fortunately, the OR1200 processor only sets a flag in the SR when a division-by-zero occurs. For the two-stage trigger, we select wires that report whether the division was signed or unsigned as trigger inputs. The attack program alternatively switches the two wires by performing signed, then unsigned division, until both analog trigger circuits are activated, deploying the attack payload. Pseudo codes for both the one-stage and two-stage attack triggering software sequences are shown in Figure 6.11 and Figure 6.12.

Triggering the attack in usermode-only code that does not alert the operating system is only the first part of a successful attack. For the second part, the attacker must be able to verify that there triggering software worked—without risk of alerting the operating system. To check whether the attack is successful, we take advantage of a special feature of some registers on the OR1200: some privileged registers are able to be read by usermode code, but the value reported has some bits redacted. We use this privilege-dependent read behavior as a side-channel to let the attacker's code know whether it has privileged access to the processor or not.

[3] Exposing the attack during normal operation may be acceptable as non-malicious software does *not* attempt to access privileged processor state. Additionally, current operating systems blindly trust the processor, so they are likely to miss sporadic privilege escalations.

```
{r0 is a non-zero register but reads as zero in user mode}
Initialize SR[0]=0                                   {initialize to user mode}
while Attack_Success==0 do
   i ← 0
   while i < 500 do
      z ← 1/0
      i ← i + 1
   end while
   if read(special register r0) ≠ 0 then
      Attack_Success ← 1
   end if
end while
```

Figure 6.11: Program that activates the single-stage attack.

```
{r0 is a non-zero register but reads as zero in user mode}
Initialize SR[0]=0                                   {initialize to user mode}
while Attack_Success==0 do
   i ← 0
   while i < 500 do
      z ← a/b                                        {signed division}
      z ← c/d                                        {unsigned division}
      i ← i + 1
   end while
   if read(special register r0) ≠ 0 then
      Attack_Success ← 1
   end if
end while
```

Figure 6.12: Program that activates the two-stage attack.

6.4.2 Analog Activity Trigger

Here we cover the implementation details of our analog triggers. To verify the first-order behavior of our analog trigger circuits, we implement, optimize, and simulate them using a SPICE simulator. Once we achieve the desired trigger behavior in simulation, we implement both the one-stage and two-stage trigger circuits in 65*nm* GP CMOS technology. Both trigger circuits are inserted into the processor to demonstrate our proposed attack. To fully characterize the performance of the trigger circuits, standalone testing structures are added to the test chip.

Implementation in 65nm GP technology

For prototype purposes, we optimize the trigger circuit towards a reliable version because we can only get a limited number of chips for measurement with no control of process variation and building a reliable circuit under process, temperature, and voltage (PVT) variations is always more challenging than only optimizing for a certain PVT range—i.e., we construct our attacks so that they work in all fabricated processors at all corner-case environments. For robustness, the Schmitt trigger shown in Figure 6.6 is used as detector circuit. Out of the three leakage paths shown in Figure 6.5, gate leakage causes most trouble because it has an exponential dependence on gate voltage, making the leakage much stronger when capacitor voltage ramps up. The gate leakage also has exponential dependence on gate oxide thickness of the fabrication technology, because gate leakage is physically quantum tunneling through gate oxide. Unfortunately, 65*nm* CMOS technology is not a favorable technology for our attack, because the gate oxide is thinner than older technologies due to dimension scaling and also thinner than latest technologies because high-κ metal gate techniques now being employed to reduce gate leakage (we use 65*nm* due to its cost savings and it is still a popular process node). Through careful sizing, it's still possible to design a circuit robust across PVT variations, but this requires trading-off *trigger time* and *retention time* as shown in in the simulation waveform of our analog activity trigger depicted in Figure 6.7.

To reduce gate leakage, another solution is to use thick oxide transistors commonly used in IO cells as the MOS cap for *Cmain*, which shows negligible gate leakage. This option provides larger space for configuration of *trigger time* and *retention time*, but requires larger area due to design rules. SPICE simulation results of the trigger circuits are shown in Figure 6.13. A zoomed waveform of the trigger operation is shown in the upper waveform, while the entire operation, including trigger and decay, is shown in the lower plot. A *trigger time* of 300*ns* and *retention time* of 25μs are marked on the waveforms. Trigger circuit using IO device is implemented for two-stage attack and the one without IO device is used for one-stage attack in the system.

We also performed exploratory simulations of our trigger circuits in 65*nm* Low Power technology, which has significantly less leakage current which is better suited for low power applications. In Low Power technology, no IO device is needed to achieve robust trigger circuits with large *trigger time* and *retention time*. Thus, from an attackers perspective, Low Power technology makes

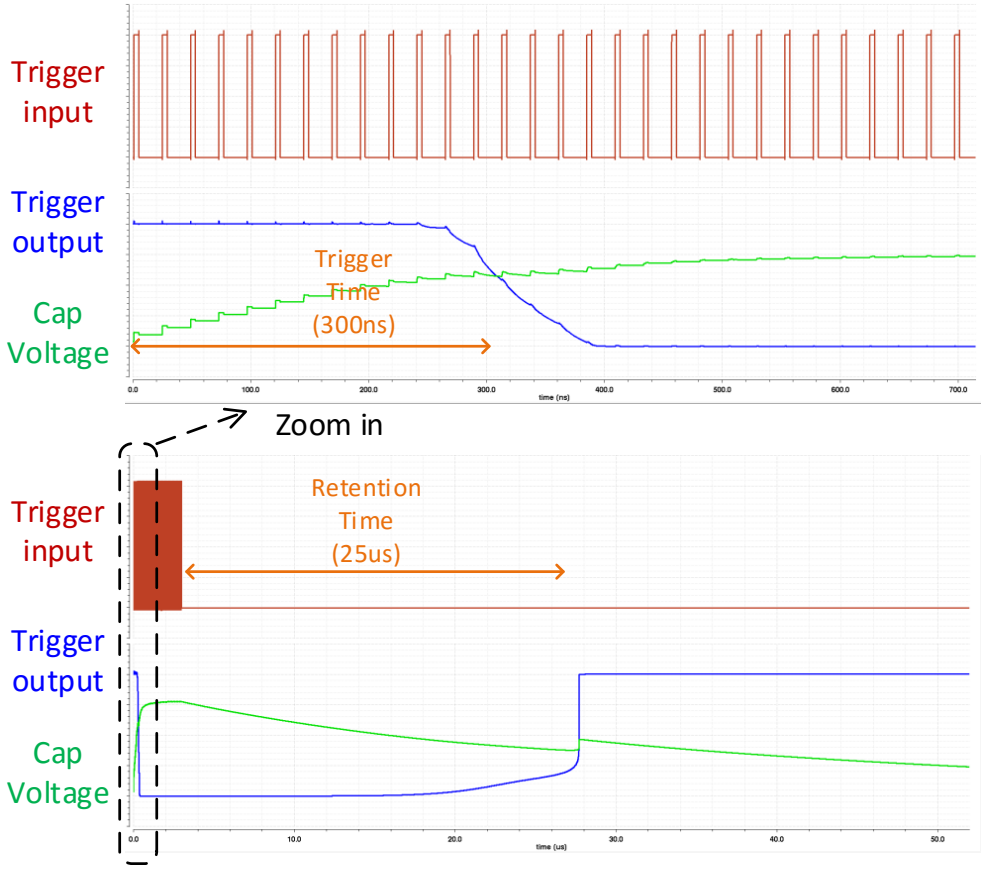


Figure 6.13: SPICE simulation waveform of analog trigger circuit using IO devices in 65nm CMOS.

implementing A2 easier and, as detailed in Section 6.5.3, harder to detect.

Inserting A2 into existing chip layouts

Since A2's analog trigger circuit is designed to follow sizing and routing constraints of standard cells and have occupy the area comparable to a single standard cell, inserting the trigger circuit to the layout at fabrication time is not complicated. All digital designs nowadays are based on standard cells and standard cells are placed in predefined rows. In typical placement and routing cases, around 60% to 70% of total area is used for standard cells, otherwise routing can not complete due to routing congestion (our chip is more challenging to attack as it has 80% area utilization). Therefore, in any layout of digital designs, empty space exists. This empty space presents an opportunity for attackers as they can occupy the free space with their own malicious circuit. In our case, we requires as little space as one cell. There are 4 steps to insert a trigger into layout of a design:

Function	Drive Strength	Width†	AC Power†	Standby Power†
NAND2	X1	1	1	1
NAND2	X4	3	3.7	4.1
NAND2	X8	5.75	7.6	8.1
DFF with Async Set	X1	6.25	12.7	2.9
DFF with Async Set	X4	7.25	21.8	6.8
DFF with Async Reset	X1	6	12.7	2.6
DFF with Async Reset	X4	7.75	21.8	7.2
DFF with Async Set and Reset	X1	7.5	14.5	3.3
DFF with Async Set and Reset	X4	8.75	23.6	8.1
Trigger w/o IO device	**-**	**8**	**7.7**	**2.2**
Trigger w/ IO device	**-**	**13.5**	**0.08**	**0.08**

* DFF stands for D Flip Flop.

† Normalized values

Table 6.1: Comparison of area and power between our implemented analog trigger circuits and commercial standard cells in 65*nm* GP CMOS technology.

1. The first step is to locate the signals chosen as trigger inputs and the target registers to attack. The insertion of A2 attack can be done at both back-end placement and routing stage and fabrication stage. Our attack model focuses on the fabrication stage because it is significantly more challenging and more stealthy compared to attack at back-end stage. The back-end stage attacker has access to the netlist of the design, so locating the desired signal is trivial. But an attack inserted at back-end stage can still be discovered by SPICE simulation and layout checks, though the chance is extremely low if no knowledge about the attack exists and given the limits of current SPICE simulators. In contrast, fabrication time attacks can only be discovered by post-silicon testing, which is believed to be very expensive and difficult to find small Trojans. To insert an attack at during chip fabrication, some insights about the design are needed, which can be extracted from layout or from a co-conspirator involved in design phase, even split manufacturing technique may not prevent the attacker from finding the target wires, as discussed in Section 6.6.3.

2. Once the attacker finds acceptable victim wires for trigger inputs and attack payload target registers, the next step is to find empty space around the victim wire and insert the analog trigger circuit. Unused space is usually automatically filled with filler cells or capacitor cells by placement and routing tools. Removing these cells will not affect the functionality or

timing, because they are inserted as the last step after all connectivity and timing checks. Because the layout of trigger circuit only uses metal 1, inserting it to unused space will not block routed signals because metal 1 is barely used for global routing.

3. To insert the attack payload circuit, the reset wire needs to be cut as discussed in Section 6.3.3. It has been shown that timing of reset signal is flexible, so the *AND* or *OR* gate only need to be placed somewhere close to the reset signal. Because the added gates can be a minimum strength cell, their area is small and finding space for them is trivial.

4. The last step is to manually do the routing from trigger input wires to analog trigger circuit and then to the payload circuits. There is no timing requirement on this path so that the routing can go around existing wires at same metal layer (jogging) or jump over existing wires by going to another metal layer (jumping), in order to ensure connection without shorting or design rule violation. If long and high metal wires become a concern of the attacker due to potential easier detection, repeaters (buffers) can be added to break long wire into small sections. Adding repeaters also reduces loading on the trigger input wire so that impacts on timing of original design is minimized. Furthermore, it is also possible that the attacker can choose different trigger input wires and/or payload according to the existing layout of the target design. This is possible because the proposed attack can be used to build variants of system level attacks.

In our OR1200 implementation, finding free space to insert the charge pump is trivial, even with the design's 80% area utilization, because the charge pump is small and there is no timing requirement on the attack circuits, affording us the freedom to distribute our attack components over a wide area of the chip. In our implementation, the distance between trigger and victim flip-flop is in near the mean of all interconnects. Connecting our attack components does require some jogging and jumping for the connections, but this is a common routing technique in commercial settings, so the information leaked by such wires is limited.

In A2, we select the victim processor and we also synthesize the chip. This means that we can bridge the semantic gap between names (and by extension functionality) at the hardware description level and traces in the mask. This level of information is representative of what back-end design house attackers would have. We also expect that it is possible for a foundry-level attacker

to implement A2. This is more difficult because a foundry-level attacker only has access to the chip layout. To accomplish the attack, the attacker must be able to identify a victim wire and to identify the victim flip-flop. Viable victim wires must have a low baseline rate of activity (given the expected use of the processor) and be controllable by the attacker to have a high enough activity to fill the trigger's capacitor. We observe that for processors, the existence of such a wire is not an issue. For the attacker to identify the wire, they must convert the chip layout back in to a purely digital representation, i.e., the structural netlist. Fortunately, this is an existing part of the back-end house design process known as Physical Verification. Thus, a foundry-level attacker can also use such a tool to obtain a netlist of the chip suitable for digital simulation. Once an attacker can simulate a chip, finding a suitable victim wire is a matter of simulating the expected workload and possible attack triggers; this is how we found viable victims for A2. Identifying the desired victim flip-flop in the generated netlist is challenging due to the lack of meaningful names. For A2, we are able to identify the victim flip-flop in a netlist with no meaningful names by writing test cases that expose the flip-flop by making it change value at a series of specific clock cycles.

Side-channel information

For the attack to be stealthy and defeat existing protections, the area, power and timing overhead of the analog trigger circuit should be minimized. High accuracy SPICE simulation is used to characterize power and timing overhead of implemented trigger circuits. Comparisons with several variants of $NAND2$ and $Dflip-flop$ standard cells from commercial libraries are summarized in Table 6.1. The area of the trigger circuit not using IO device is similar to a X4 strength $Dflip-flop$. Using an IO device increases trigger circuit size significantly, but area is still similar to the area of 2 standard cells, which ensures it can be inserted to empty space in final design layout. AC power is the total energy consumed by the circuits when input changes, the power numbers are simulated by doing SPICE simulation on a netlist with extracted parasitics from our chip layout. Standby power is the power consumption of the circuits when inputs are static and comes from leakage current of CMOS devices.

In A2, the analog trigger circuit is directly feeds off of the victim wire, which is the only part in the attack that creates a timing disturbance to the original design. Before and after inserting the A2, we extract parasitics from the layouts to do high accuracy simulation of the victim wire's

delay. Results show that rising and falling delay before trigger insertion are $19.76ps$ and $17.18ps$ while those after trigger insertion are $20.66ps$ and $18.45ps$. Extra delay is $1.2ps$ on average, which is the timing overhead of the attack. $1.2ps$ is only 0.33% of $4ns$ clock period and well below the process variation and noise range. Besides, in practical measurement, $1.2ps$ is nearly impossible to measure. unless high resolution time to digital converter is included on chip, which is impractical due to its large area and power overhead.

Comparison to digital-only attacks

If we look at a previously proposed, digital only and smallest implementation of a privilege escalation attack [51], it requires 25 gates and $80\mu m^2$ while our analog attack requires as little as one gate for the same effect. Our attack is also much more stealthy as it requires dozens of consecutive rare events, where the other attack only requires two. We also implement a digital only, counter-based attack that aims to mimic our A2. The digital version of A2 requires 91 cells and $382\mu m^2$, almost two orders-of-magnitude more than the analog counterpart. These results demonstrate how analog attacks can provide attackers the same power, control, and more stealthiness as existing digital attacks, but at a much lower cost.

Trigger characterization

To fully characterize the fabricated trigger circuit, a standalone testing structure as shown in Figure 6.14 is included in the test chip. A digital clock divider and duty cycle controller takes parameters from the scan chain to generate a simulated victim wire for the trigger. A feedback loop connected to an *AND* gate is used to stop the trigger input when the trigger output is activated. A counter counts the number of transitions of the trigger input. It stops when the trigger output is activated. The counter value is read out using the scan chain. Combining the count, clock frequency and clock divider ratio (i.e., the toggle rate of the victim wire), we can calculate the *trigger time*. After the trigger activates and victim wire stops toggling due to the *AND* gate, the capacitor voltage will slowly leak away until the trigger is deactivated. Once it is deactivated, the counter will restart. By taking readings fast, we can roughly measure the time interval between counter stops and restarts, which is the *retention time* of the trigger circuit.

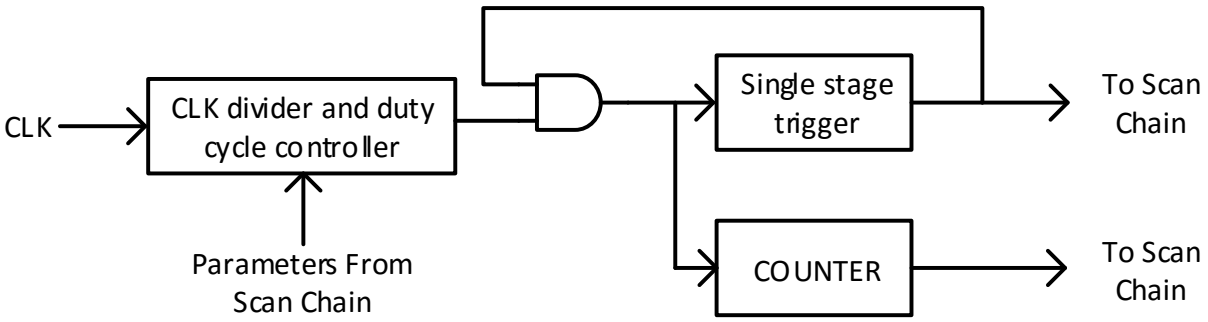


Figure 6.14: Testing structure to characterize the *trigger time* and *retention time* of implemented analog trigger circuits.

6.5 Evaluation

We perform all experiments with the fabricated $2.1mm^2$ malicious OR1200 processor in $65nm$ CMOS technology. Figure 6.15 shows this processor, including where the different functional blocks are located within the processor. Figure 6.15 also shows where we add A2, with two levels of zoom to aide in understanding the challenge of identifying A2 in a sea of non-malicious logic. In fact, A2 occupies less than 0.08% of the chip's area. Our fabricated chip actually contains two sets of attacks: the first set of attacks are one and two-stage triggers baked-in to the processor that we use to assess the end-to-end impact of A2. The second set of attacks exist outside of the processor and are used to fully characterize A2's operation.

We use the testing setup shown in Figure 6.16 to evaluate our attacks' response to changing environmental conditions and a variety of software benchmarks. The chip is packaged and mounted on a custom testing PCB to interface with personal computer. We use the LabVIEW program to control a digital interface card that reads and writes from the chip through a custom scan chain interface. The scan chain interface enables us to load programs to the processor's memory and also to check the values of the processor's registers. The testing board is kept in a temperature chamber to evaluate our attacks under temperature variations. To clock the processor, we use an on-chip clock generator that generates a $240MHz$ clock at the nominal condition ($1V$ supply voltage and $25°C$). We use a programmable clock divider to convert the $240MHz$ clock into the desired clock frequency for a given experiment.

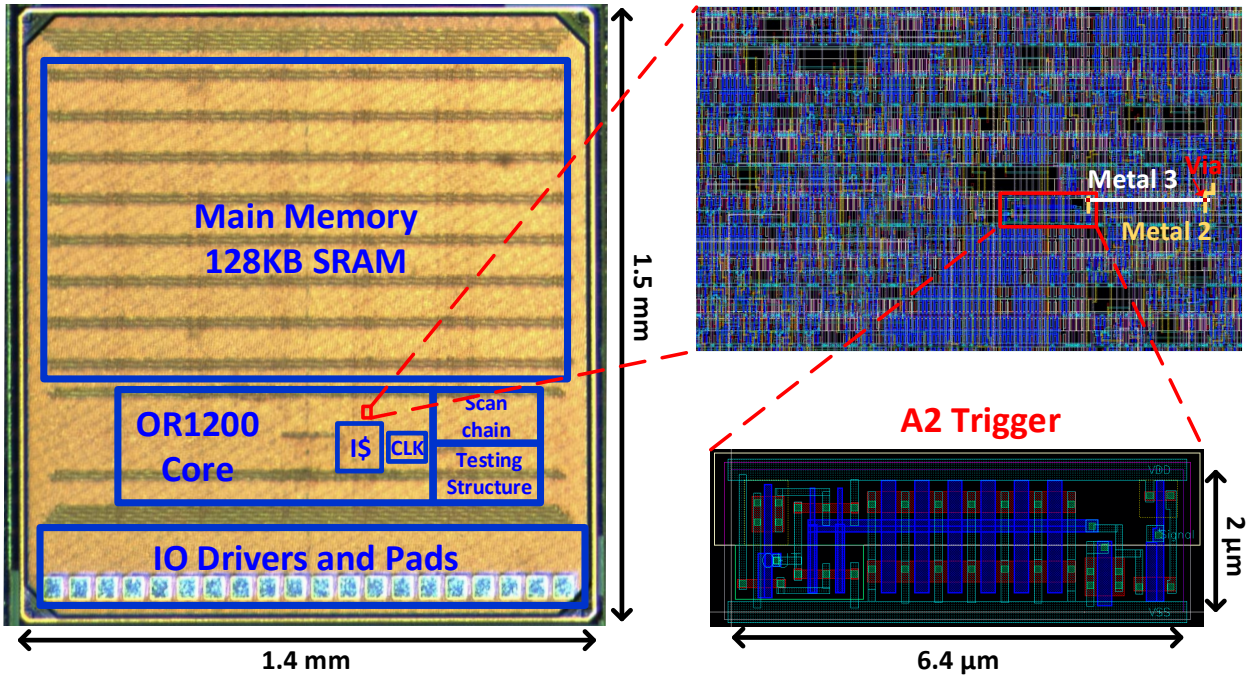


Figure 6.15: Die micrograph of analog malicious hardware test chip with a zoom-in layout of inserted A2 trigger.

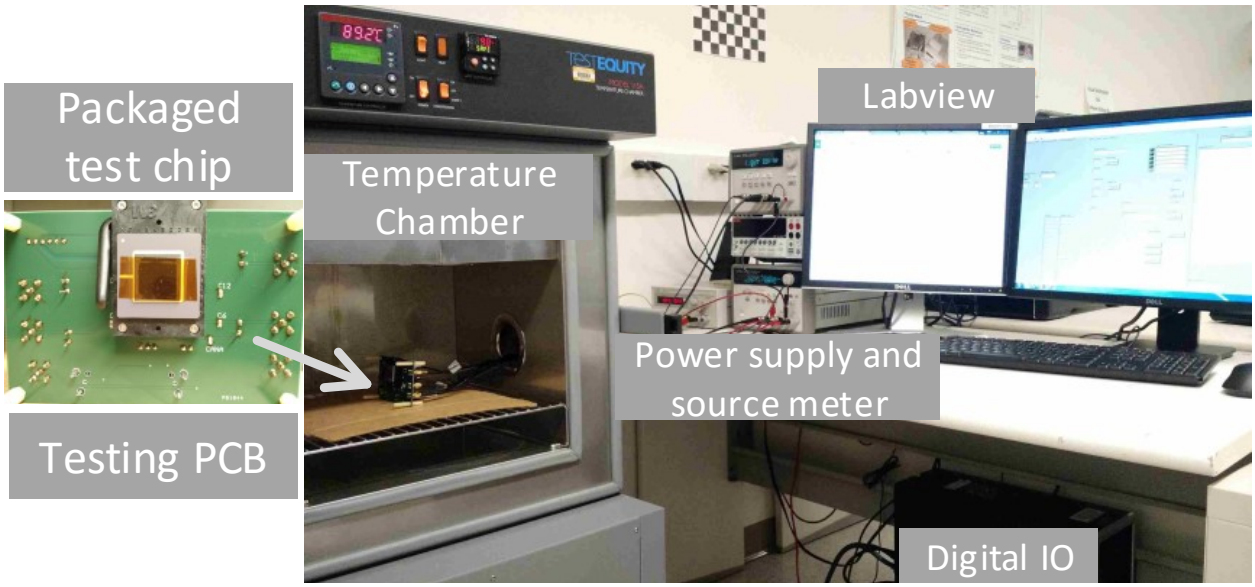
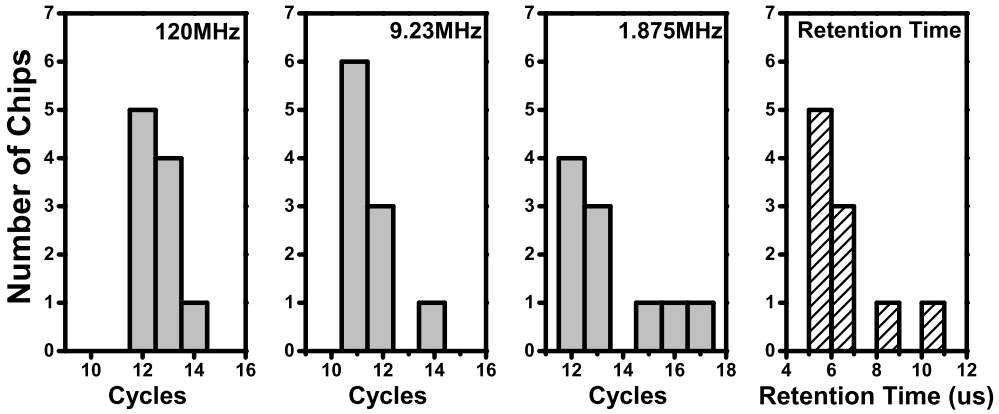
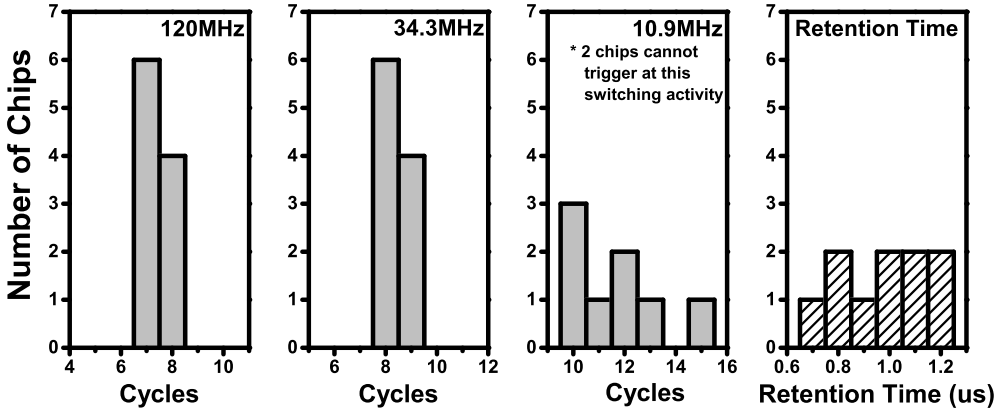


Figure 6.16: Testing setup for test chip measurement.



(a) Distribution of analog trigger circuit using IO device



(b) Distribution of analog trigger circuit using only core device

Figure 6.17: Measured distribution of retention time and trigger cycles under different trigger input divider ratios across 10 chips at nominal 1V supply voltage and 25°*C*.

6.5.1 Does the Attack Work?

To prove the effectiveness of A2, we evaluate it from two perspectives. One is a system evaluation that explores the end-to-end behavior of our attack by loading attack-triggering programs on the processor, executing them in usermode, and verifying that after executing the trigger sequence, they have escalated privilege on the processor. The other perspective seeks to explore the behavior of our attacks by directly measuring the performance of the analog trigger circuit, the most important component in our attack, but also the most difficult aspect of our attack to verify using simulation. To evaluate the former, we use the in-processor attacks and for the later, we use the attacks implement outside the processor with taps directly connected to IO pins.

System attack

Malicious programs described in Section 6.4.1 are loaded to the processor and then we check the target register values. In the program, we initialize the target registers $SR[0]$ (the mode bit) to user mode (i.e., 0) and $SR[1]$ (a free register bit that we can use to test the two-stage trigger) to 1. When the respective triggers deploys the attack, the single-stage attack will cause $SR[0]$ to suddenly have a 1 value, while the two-stage trigger will cause $SR[1]$ to have a 0 value—the opposite of their initial values. Because our attack relies on analog circuits, environmental aspects dictate the performance of our attack. Therefore, we test the chip at 6 temperatures from $-25^{\circ}C$ to $100^{\circ}C$ to evaluate the robustness of our attack. Measurement results confirm that both the one-stage and two-stage attacks in all 10 tested chips successfully overwrite the target registers at all temperatures.

Analog trigger circuit measurement results

Using the standalone testing structure shown in Figure 6.14, *number of cycles until trigger* and *retention time* can be characterized. We use the $240MHz$ on-chip clock to simulate the toggling of a victim wire that feeds the trigger circuits under test. To show how our attack triggers respond to a range of victim activity levels, we systematically sweep clock division ratios which simulates a similar range of victim wire activities.

Figure 6.17 shows the measured distribution of *retention time* and trigger cycles at 3 different trigger toggle frequencies across 10 chips. The results show that our trigger circuits have a regular behavior in the presence of real-world manufacturing variances, confirming SPICE simulation results. *retention time* at the nominal condition ($1V$ supply voltage and $25^{\circ}C$) is around $1\mu s$ for trigger with only core devices and $5\mu s$ for attacks constructed using IO devices. Compared to SPICE simulation results, in Figure 6.7 and Figure 6.13, trigger without IO devices has close results while trigger with IO device shows 4 times smaller retention time than simulations suggest. This is reasonable because gate leakage of IO devices is negligible in almost any designs and the SPICE model is a rough estimation. Table 6.2 provides the number of cycles until triggering for both trigger circuits (i.e., with and without IO devices) from fabricated chip measurements and SPICE simulations to validate the accuracy of simulation. An attacker wants the simulator to be as

Trigger Circuit	Toggle Rate (MHz)	Measured (10 chip avg)	Simulated (Typical corner)
w/o IO device	120.00	7.4	7
w/o IO device	34.29	8.4	8
w/o IO device	10.91	11.6	10
w/ IO device	120.00	12.6	14
w/ IO device	9.23	11.6	13
w/ IO device	1.88	13.5	12

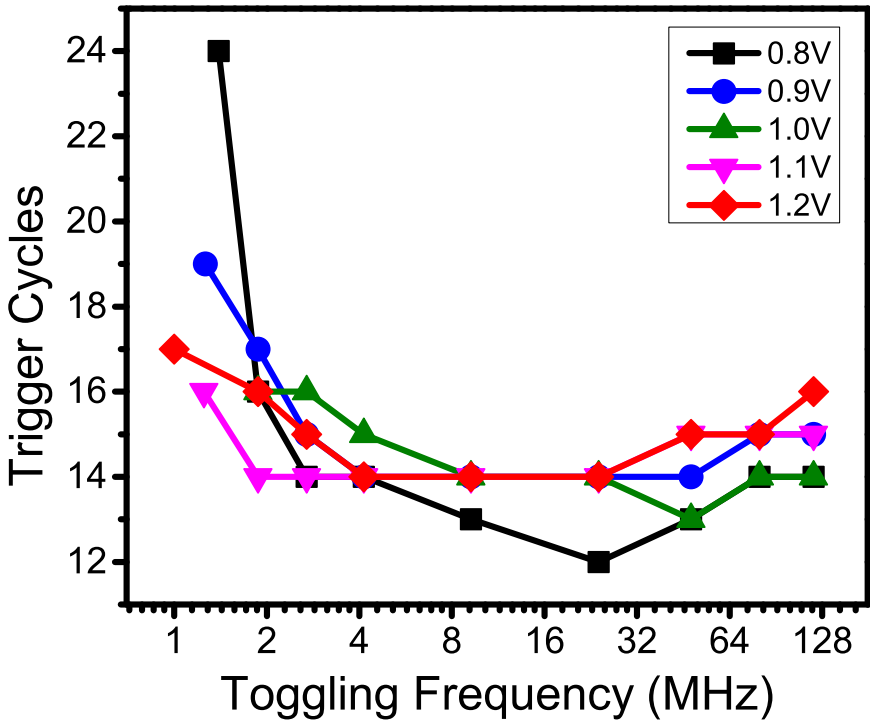
Table 6.2: Comparison of how many cycles it takes to activate fully the trigger for our fabricated chip (Measured) and for HSPICE (Simulated) versions of our analog trigger circuit.

accurate as possible as the cost and time requirement of fabricating test chips make it impractical to design analog attacks without a reliable simulator. Fortunately, our results indicate that SPICE is capable at providing results of sufficient accuracy for these unusual circuits based on leakage currents.

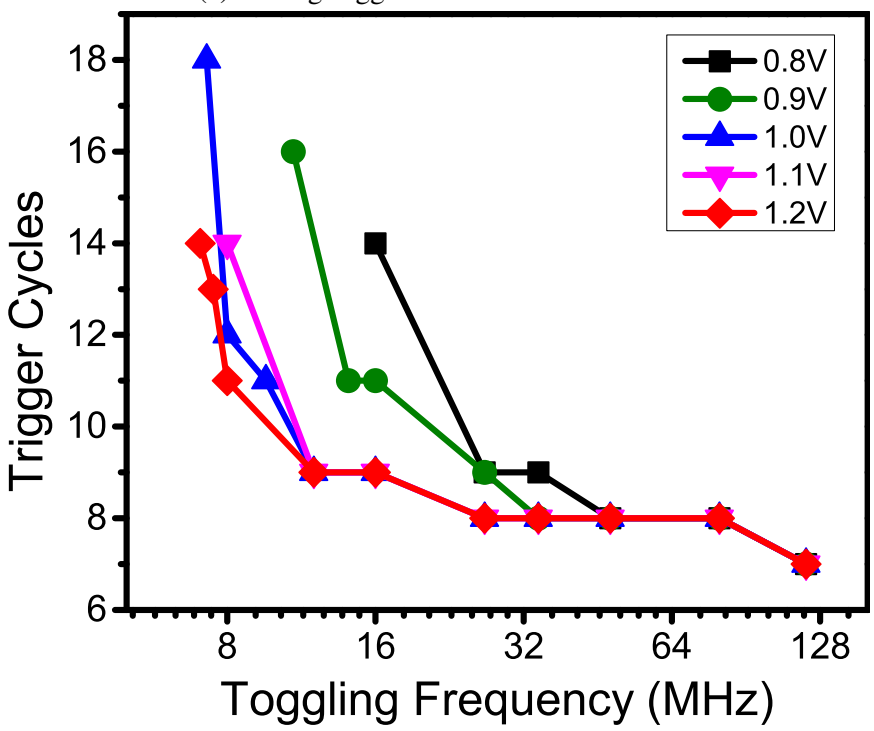
To verify the implemented trigger circuits are robust across voltage and temperature variations (as SPICE simulation suggests), we characterize each trigger circuit under different supply voltage and temperature conditions. Figure 6.18 and Figure 6.19 show how many cycles it takes (on average) for each trigger circuit to activate fully when the simulated victim wires toggles between $.46MHZ$ and $120MHz$, when the supply voltage varies between $0.8V$ and $1.2V$, and when the ambient temperature varies between $-25°C$ and $100°C$.

As expected, different conditions yield different minimum toggling rates to activate the trigger. It can be seen that temperature has a stronger impact on our trigger circuit's performance because of leakage current's exponential dependence on temperature. At higher temperature, more cycles are required to trigger and higher switching activity is required because leakage from capacitor is larger. The exception to this happens with the trigger constructed using IO devices, at very low temperature. In this case, leakage currents are so small that the change in trigger cycles comes mainly from the setup time of Schmitt trigger, higher toggling inputs spend more cycles during the setup time. SPICE simulation predicts these results as well.

Lastly, once the trigger activates, it will only remain in the activated state for so long, barring continued toggling from the victim wire. The window of time that a trigger stays activated is critically important for series-connected multi-stage trigger circuits. This window is also controlled by manufacturing variances and environmental conditions. Variation of retention time across $-25°C$

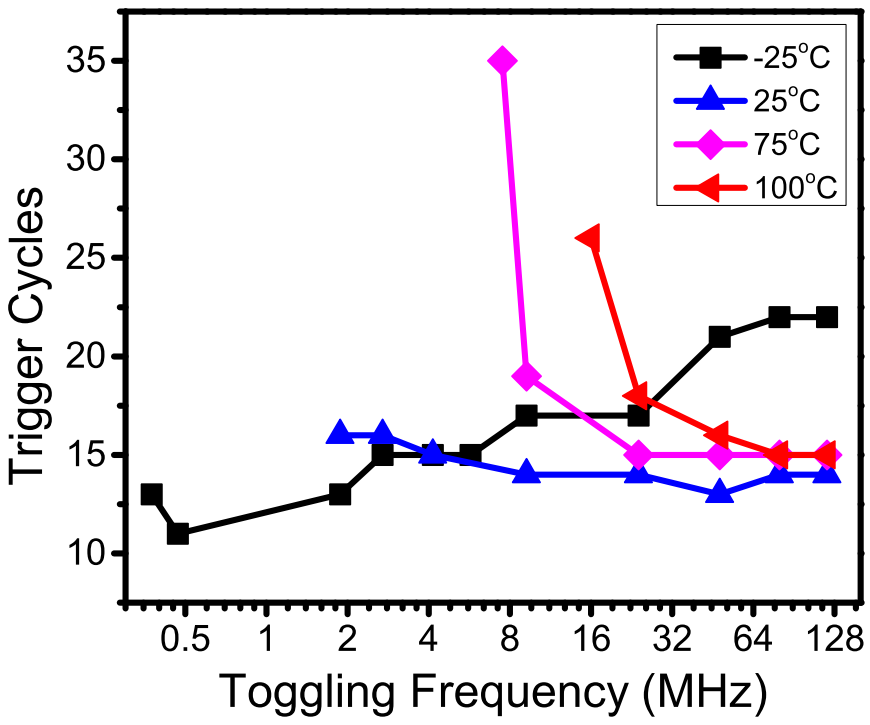


(a) Analog trigger circuit with IO device

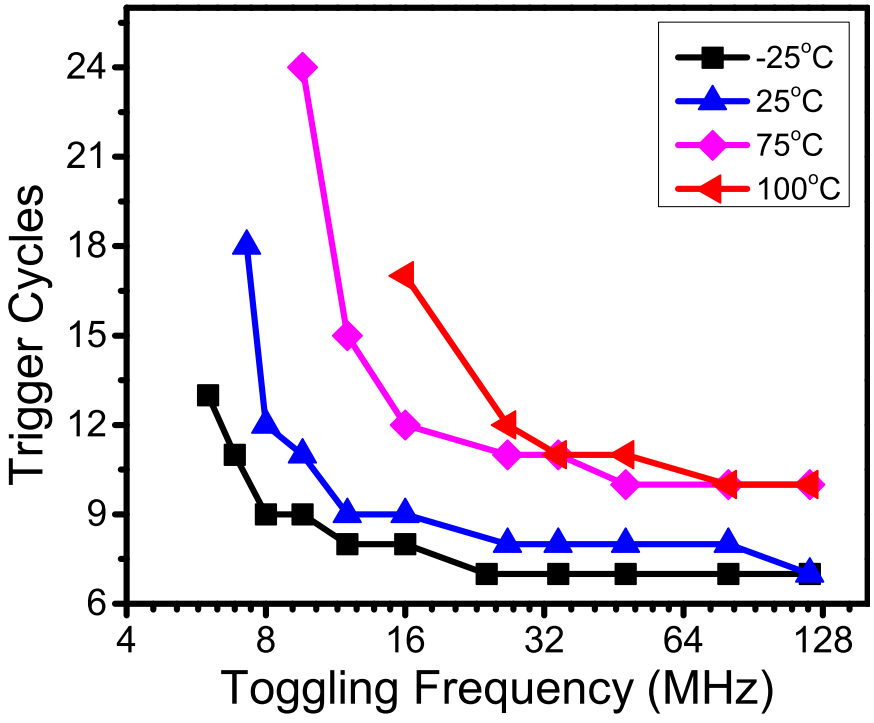


(b) Analog trigger circuit with only core device

Figure 6.18: Measured trigger cycles under different input frequency at different supply voltages.



(a) Analog trigger circuit with IO device



(b) Analog trigger circuit with only core device

Figure 6.19: Measured trigger cycles under different input frequency at different ambient temperatures.

to 100°C is plotted in Figure 6.20, which shows that the retention time of both trigger circuits is long enough to finish the attack across wide temperature range. Trigger circuits constructed with IO devices have a larger dependence on temperature because of different temperature dependencies for different types of devices. The variation of *cycles until triggering* and *retention time* across PVT variations implies the possibility that an attacker can include the environmental condition as part of the trigger. For example, a low activity trigger input can only trigger the attack at low temperatures according to the measurement results; great news if you want your attack to work in the North Pole, but not the tropics. Attackers can also tune the circuits towards stricter requirement to trigger so that the attack is never exposed at higher temperatures to further avoid detections.

6.5.2 Is the Attack Triggered by Non-malicious Benchmarks?

Another important property for any hardware Trojan is not exposing itself under normal operations. Because A2's trigger circuit is only connected to the trigger input signal, digital simulation of the design is enough to acquire the activity of the signals. However, since we make use of analog characteristics to attack, analog effects should also be considered as potential effects to accidentally trigger the attack. Therefore, we ran 5 selected programs from the MiBench embedded systems benchmark suite. We select MiBench [48] because it targets the class of processor that best fits the OR1200 and it consists of a set of well-understood applications that are popular system performance benchmarks in both academia and in industry. MiBench consists of 31 applications, spread across 6 resource-usage-based classes. To validate that A2's trigger avoids spurious activations from a wide variety of software, we select 5 benchmark applications from MiBench, each from a different class. This ensures that we thoroughly test all subsystems of the processor—exposing likely activity rates for the wires in the processor. Again, in all programs, the victim registers are initialized to opposite states that A2 puts them in when its attack is deployed. The processor runs all 5 programs at 6 different temperatures from $-25°C$ to $100°C$. Results prove that neither the one-stage nor the two-stage trigger circuit is exposed when running these benchmarks across such wide temperature range.

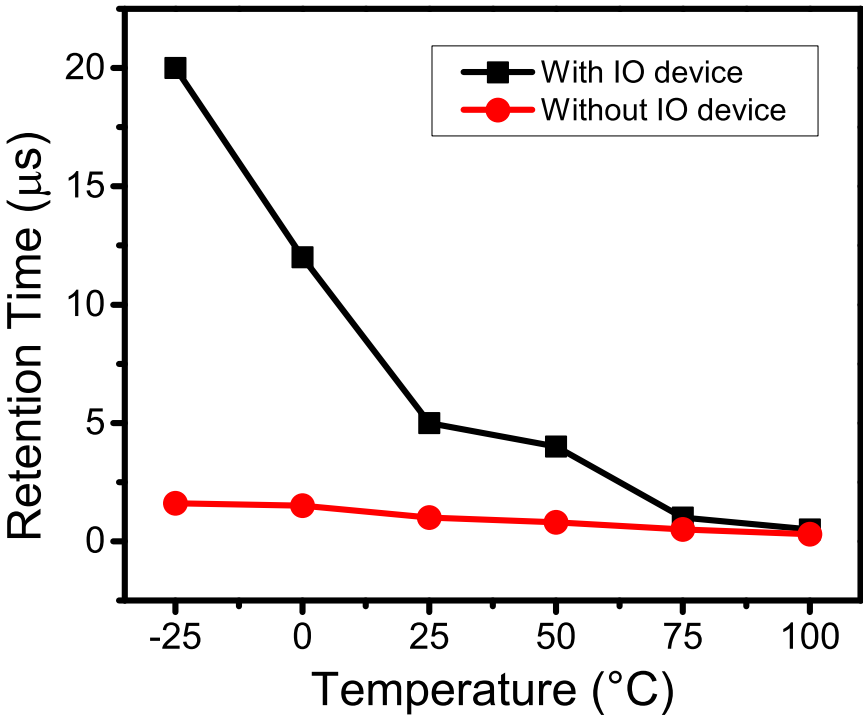


Figure 6.20: Measured retention time of analog trigger circuits across temperatures.

6.5.3 Existing Protections

Existing protections against fabrication-time attacks are mostly based on side-channel information, e.g., power, temperature, and delay. In A2, we only add one gate in the trigger, thus minimizing power and temperature perturbations caused by the attack.

Table 6.3 summarizes the total power consumption measured when the processor runs our five benchmark programs, at the nominal condition ($1V$ supply voltage and $25°C$). A Keithley 2400 sourcemeter is used to measure the power consumption of the processor, which can measure down to $1\mu A$ in our measurement range. All the values in Table 6.3 are average values across the entire program execution. The variation of power consumption in all cases are limited to $\pm 3\mu W$. Direct measurement of trigger circuit power is infeasible in our setup, so simulation is used as an estimation. It was shown earlier that SPICE model matches measurement results in terms of trigger performance. Simulated trigger power consumption in Table 6.1 translates to $5.3nW$ and $0.5\mu W$ for trigger circuits constructed with and without IO devices. These numbers are based on the assumption that trigger inputs keep toggling at 1/4 of the clock frequency of $240MHz$, which is the maximum switching activity that our attack program can achieve on the selected victim wire.

Program	Power (mW)
Standby	6.210
Basic math	23.703
Dijkstra	16.550
FFT	18.120
SHA	18.032
Search	21.960
Single-stage Attack	19.505
Two-stage Attack	22.575
Unsigned Division	23.206

Table 6.3: Power consumption of our test Chip running a variety of benchmark programs.

In the common case of non-attacking software, the switching activity is much lower—approaching zero—and only lasts a few cycles so that the extra power due to our trigger circuit is even smaller. In our experiments, the power of the attack circuit is orders-of-magnitude less than the normal power fluctuations that occur in a processor while it executes different instructions.

Besides side-channel information leaked by attack circuit itself, parasitic impacts of attack circuits on original design should also be considered. Adding new transistors around existing ones introduces negligible differences to the existing devices, because manufacturing steps like doping, lithography, and planarization are well controlled in modern CMOS IC manufacturing through the use of dummy doping/poly/metal fill. This fill maintains a high density of materials over large windows. The tiny inserted A2 trigger will not significantly change the overall density in a window and therefore does not cause systematic device variations. Besides, isolation between transistors avoids their coupling.

Coupling between malicious and original wires may cause cross-talk and more delay to the original wires. However, in CMOS manufacturing, the metal fill step adds floating metal pieces to empty spaces in the chip layout so that the unit parasitic capacitance of all wires are similar. An attacker can limit cross-talk effects through careful routing to avoid long parallel wires.

6.6 Discussion

Now that we know A2 is an effective and stealthy fabrication-time attack, we look forward to possible defenses, including a discussion of the impact of split manufacturing and 3D-IC on our attacks. Before delving into defending against A2, we qualitatively address the challenge of implementing an A2-like attack in x86 processors.

6.6.1 Extending A2 to x86

We implement A2 on the OR1200 processor because it is open source. While the OR1200 processor is capable enough to run Linux, its complexity is closer to a mid-range ARM part, far below that of modern x86 processors from Intel and AMD. A natural question is if our attack technique applies to x86 processors and, if so, how does the attack's complexity scale with processor complexity.

We expect a A2-like attack in x86 processors to be much harder to detect and easier to implement than its OR1200 counterpart. While there are more viable victim registers in x86, A2 still only needs to target a single register to be effective. Also, A2's overhead comes primarily from its trigger circuit, but the complexity of the trigger is much more dependent on how "hidden" the attacker wants the attack to be than on the complexity of the processor. In fact, we expect that there are far more viable victim wires (highly-variable and controllable activity) due to the internal structure of complex, out-of-order processors like the x86. The only aspect of scaling to an x86-class processor that we anticipate as a challenge is maintaining controllability as there are many redundant functional units inside an x86, so a trigger would either need to tap equivalent wires in all functional units or be open to some probabilistic effects.

6.6.2 Possible Defenses

There are a few properties that make our attacks hard to detect: 1) we require adding as little as a single gate 2) our attack has a sophisticated trigger and 3) our trigger works in the analog domain, gradually building charge until it finally impacts the digital domain. Given these properties, defenses that measure side-channel information (e.g., current and temperature) have little

hope of detecting the impact of a single gate in a sea of 100,000 gates. The same holds true for defenses that rely on visual inspection. Even if a defender were to delayer a chip and image it with a scanning electron microscope, our malicious gate is almost identical to all the other gates in a design. One option might be to focus the search on the area of the chip near security-critical state holding flip-flops.

If it is impractical to expect defenders to visually identify our attacks or to be able to detect them through measuring current or temperature, what about testing? One of the novel features of A2 is the trigger. In our implementation (Section 6.4), we carefully design the trigger to make it extremely unlikely for unknowing software—including validation tests—to trigger the attack. In fact, we built a trigger so immune to unintended activations that we had to employ sleds of inline assembly to get an activity ratio high enough to trigger our attack. This indicates that anything short of comprehensive testing is unlikely to expose the trigger [4].

Given that post-fabrication testing is unlikely to expose our attack and our attack's impact on known side-channels is buried within the noise of a circuit, we believe that a new type of defense is required: we believe that the best method for detecting our attack is some form of runtime verification that monitors a chip's behavior in the digital domain.

6.6.3 Split Manufacturing

One promising future defense to malicious circuits inserted during fabrication is split manufacturing [53–56] and 3D-IC [57]. The idea behind defenses incorporating split manufacturing is to divide a chip into two parts, with one part being fabricated by a cheap, but untrusted, fabrication house, while the other part gets fabricated by an expensive, but trusted, fabrication house (that is also responsible for putting the two parts together in a single chip). The challenge is determining how to divide the chip into two parts.

One common method is to divide the chip into gates and wires [57]. The idea behind this strategy is that by only moving a subset of wires to the trusted portion of the chip, it will be cheaper to fabricate at the trusted fabrication house, while serving to obfuscate the functionality from the untrusted fabrication house. This obfuscation makes it difficult for an attacker to determine which

[4] Even if test cases somehow activated our attack, the onus is on the testing routines to catch our malicious state change. Observe that most non-malicious software runs the same regardless of privilege level.

gates and wires (of the ones they have access to) to corrupt.

From this description, it might seem as if current split manufacturing-based defenses are a viable approach to stopping A2. This is not the case as A2 changes the state in a flip-flop, but only wires are sent to the trusted fabrication house. Future split manufacturing approaches could move a subset of flip-flops to the trusted part of the chip along with a subset of wires, but that increases the cost of an already prohibitively expensive defense. Additionally, recent research shows that even when a subset of wires are missing, it is possible to reverse engineer up to 96% of the missing wires using knowledge of the algorithms used in floor-planning, placement, and layout tools [53]. In fact, we already take advantage of some of this information in identifying the victim wire that drives our trigger circuit and in identifying the victim flip-flop.

Previous works [54, 55] also proposed splitting manufacturing at low-level metal layers, even down to lowest metal layer. Splitting at metal 1 is a potentially effective method to defend against A2 attack if carried out by untrusted manufacturer. However, this approach introduces an extremely challenging manufacturing problem due to the small dimension of low-level metal layers and tremendous amount of connections to make between two parts, not to mention the expense to develop a trusted fabrication house with such capabilities. There has been no fabricated chips demonstrating that such a scheme works given the constraints of existing manufacturers.

6.7 Related Work

A2 is a fabrication-time attack. There is almost 10 years of previous work on fabrication-time attacks and defenses. In this section, we document the progression of fabrication-time attacks from 100-gate circuits targeted at single-function cryptographic chips, aimed at leaking encryption keys to attacks that work by corrupting existing gates aimed at more general functionality. The historical progression of fabrication-time attacks highlights the need for a small, triggered, and general-purpose attack like A2.

Likewise, for fabrication-time defenses, we document the progression of defenses from complete chip characterization with a heavy reliance on a golden reference chip to defenses that employ self-referencing techniques and advance signal recovery (removing the requirement of a golden chip). We conclude with defenses that move beyond side-channels, into the real of on-chip sensors

aimed at detecting anomalous perturbations in circuit performance presumably due to malicious circuits. The historical progression of fabrication-time attack defenses shows that while they may be effective against some known attacks, there is a need for a new type of defense that operates with more semantic information.

6.7.1 Fabrication-time Attacks

The first fabrication-time hardware attack was the addition of 100 gates to an AES cryptographic circuit aimed at creating a side-channel that slowly leaks the private key [58]. The attack circuit works by modulating its switching activity (i.e., increasing or decreasing the attack's current consumption) in a way that encodes the secret key on the current consumed by the chip as a whole. This method of attack has four disadvantages: 1) the attack has limited scope 2) the attacker must have physical access to the victim device and 3) the attack is always-on, making it more detectable and uncontrollable. To mute their attack's effect on the circuit, the authors employ a spread-spectrum technique to encode single bits of the key on many clock cycles worth of the power trace of the device. This technique helps conceal the attack from known, side-channel based, fabrication-time defenses at the cost of increased key recovery time.

Another fabrication-time method for creating malicious circuits is to modify the fabrication process so that natural process variation is shifted outside the specified tolerances. Process reliability Trojans [59] show how an attacker can cause reductions in reliability by accelerating the wearing out mechanisms for CMOS transistors, such as Negative Bias Temperature Instability (NBTI) or Hot Carrier Injection (HCI). Process reliability Trojans affect an entire chip and affect some chips more than others (the effect is randomly distributed the same way as process variation); the goal is to cause the entire chip to fail early. While the paper does not implement a process Trojan, the authors explore the knobs available for implementing a process reliability Trojan and discuss the theory behind them. The value of this attack is that it is very difficult to detect as a defender would have to destructively measure many chips to reverse-engineer the fabrication parameters. A2 represents a different design point: a targeted attack that is controllable by a remote attacker.

A targeted version of a process reliability Trojan is the dopant-level Trojan [35]. Instead of

adding additional circuitry to the chip (e.g., the side-channel Trojan) or changing the properties of the entire chip (e.g., the process reliability Trojan), dopant-level Trojans change the behavior of existing circuits by tying the inputs of logic gates to logic level 0 or logic level 1. By carefully selecting the logic value and the gates attacked, it is possible to mutate arbitrary circuits into a malicious circuit. This approach is incredibly stealthy because there are no extra gates or wires, but comes with limitations. First, while there are no extra gates or wires added for the attack, more recent work shows that removing additional layers (down to the contact layers) of the chip reveals the added connections to logic 0 and logic 1 [37]. Note that removing these extra layers and imaging the lower layers is estimated to be 16-times more expensive than stopping at the metal layers. A second limitation is that the attacker can only modify existing circuits to implement their attack. This makes it difficult to construct attack triggers resulting in an exposed attack payload—making detection more likely. Recent defenses seek to prevent dopant-level attacks by obfuscating the circuit and using split manufacturing [57]. A2 trades-off some detectability in the metal layers of the chip for less detectability by testing. The observation driving this is that every chip has its functionality tested after fabrication, but it is prohibitively expensive to delayer a chip and image it with a scanning electron microscope. By using analog circuits, A2 makes it possible to implement complex attack triggers with minimal perturbations to the original circuit.

The most recent fabrication-time attack is the parametric Trojans for fault injection [36]. Parametric Trojans build on dopant-level Trojan by adding some amount of controllability to the attack. Parametric Trojans rely on power supply voltage fluctuations as a trigger. For example, imagine a dopant-level attack that only drives the input of a logic gate to 1 or 0 when there is a dip in the supply voltage. Because this requires that the attacker has access to the power supply of a device, the goal is to facilitate fault-injection attacks (e.g., erroneous result leaks part of the key as in RSA attacks [60]).

6.7.2 Fabrication-time Defenses

There are three fundamental approaches to defend against fabrication-time malicious circuits: 1) side-channel-based characterization 2) adding on-chip sensors and 3) architectural defenses. This section covers example defenses that use each approach and qualitatively analyze how A2

fares against them.

Side-channels and chip characterization

IC fingerprinting [38] is the first attempt to detect malicious circuits added during chip fabrication. IC fingerprinting uses side-channel information such as power, temperature, and electromagnetic measurements to model the run time behavior of a golden (i.e., trusted) chip. To clear untrusted chips of possible malice, the same inputs are run on the suspect chip and the same side-channel measurements collected. The two sets of measurements are then compared, with a difference above a noise threshold causing the chip to be labeled as malicious. The more golden chips available, the better the noise modeling. IC fingerprinting works well when there are a set of trusted chips, the chip's logic is purely combinational, and it is possible to exercise the chip with all possible inputs. The authors also point out that their approach requires that Trojans be at least .01% of the circuit; in A2 the Trojan is an order of magnitude smaller than that—not to mention that we attack a processor.

Another side-channel-based approach is to create a path delay fingerprint [39]. This is very similar to IC fingerprinting, except with a heavier reliance on the chip being purely combinational. To create a path delay fingerprint, testers exercise the chip with all possible test cases, recording the input-to-output time. The observation is that only malicious chips will have a path delay outside of some range (the range depends on environmental and manufacturing variances). Even if it is possible to extend this approach to sequential circuits and to meaningfully train the classifier where comprehensive testing is impractical, A2 minimizes the impacts on the delay of the surrounding circuit to hide into environmental variation and noise (Section 6.4.2) and the attack modifies state directly.

Building from the previous two defenses is gate-level characterization [40]. Gate-level characterization is a technique that aims to derive characteristics of the gates in a chip in terms of current, switching activity, and delay. Being a multi-dimensional problem, the authors utilize linear programming to solve a system of equations created using non-destructive measurements of several side-channels. A2 evades this defense because it operates in the analog domain.

Electromagnetic fingerprinting combined with statistical analysis provides a easier approach to measure local side-channel information from small parts of a chip and suppress environmental

impacts [61]. Because EM radiation from A2 only occurs when the attack is triggered, it evades defenses that assume EM signals are different in attacked designs even if the Trojan is dormant.

One major limitation of characterization-based defenses is the reliance on a golden reference chip. TeSR [41] seeks to replace a golden chip with self-referencing comparisons. TeSR avoids the requirement of a golden chip by comparing a chip's transient current signature with itself, but across different time windows. Besides eliminating the need for a golden chip, TeSR also enables side-channel techniques to apply to more complex, sequential circuits. Unfortunately, TeSR requires finding test cases that activate the malicious circuit to be able to detect it. While TeSR may work well against dopant-level Trojans, we include a complex trigger in A2 that avoids accidental activations. Additionally, results in Section 6.4 suggest that the assumption underlying TeSR—that malicious and non-malicious side-channel measurements are separable—is not true for A2-like attacks.

Adding on-chip sensors

As mentioned, using side-channel information to characterize chip delay is limited to combinational circuits. One defense suggests measuring delay locally through the addition of on-chip sensors [42]. The proposed technique is able to measure precisely the delay of a group of combinational paths—these paths could be between registers in a sequential circuit. Much like in the side-channel version, the sensors attempt to characterize the delay of the monitored paths and detect delays outside an acceptable range as potential malice. The increased accuracy and control over the side-channel version comes at the cost of added hardware: requires the addition of a shadow register for every monitored combinational path in the chip and a shadow clock that is a phase offset version of the main clock. A comparator compares the main register and the shadow register, with a difference indicating that the combinational delay feeding the main register has violated its setup requirement. This approach is similar to Razor [62], but here the phase shift of the shadow clock is gradually adjusted to expose changes in delay. A2 avoids this defense because it modifies processor state directly, not affecting combinational delays.

Adding to the list of tell tale features is Temperature Tracking [43]. Temperature Tracking uses on-chip temperature sensors to look for temperature spikes. The intuition is that when malicious hardware activates, it will do so with an unusually high (and moderate duration) burst of activity.

The activity will increase current consumption, that then produces temperature increases. Unfortunately, results from Section 6.5 show that this intuition is invalid for our malicious processor. A2 is a single gate in a sea of 100,000 gates, so its current consumption is muted. Also, A2's trigger gradually builds charge and the payload lasts for a very short duration not able to be capture at the slow rate of thermal variation. In general, it is possible for other attackers to hide their attacks from this approach by placing their malicious circuits in an active area of the chip, or by making their attack infrequently active and active for short durations.

The most recent on-chip sensor proposal targeted at detection malicious circuits added during fabrications hearkens back to IC fingerprinting in that the goal is to monitor the power rails of the chip [44]. The authors propose adding power supply monitoring sensors that detect fluctuations in a supply's characteristic frequencies. As has been noted with previous approaches, our results show that there are cases where there is *no* difference in power supply activity between the case where the malicious circuit is active versus inactive.

A2 defeats defenses that rely on characterizing device behavior through power, temperature, and delay measurements by requiring as few as one additional gate and by having a trigger that does not create or destroy charge, but redirects small amounts of charge. In addition, A2's analog behavior means that *cycle-to-cycle changes are small, eventually accumulating to a meaningful digital change*.

Eliminating unused space

BISA [63] is a promising defense against fabrication-time attacks that seeks to prevent attackers from adding components to a design by eliminating all empty space that could be used to to insert attack logic. A perfect deployment of BISA does indeed make implementing A2 more challenging. Unfortunately, the small area of A2 presents a challenging problem to any BISA implementation, because *all* empty space must be filled by BISA cells with no redundant logic or buffers—as an attacker can replace these with their attack circuit and the behavior of the design remains. Also, a perfect BISA implementation requires 100% test coverage—an impractical requirement, otherwise an attacker can replace logic not covered in the tests. In addition, implementing BISA significantly reduces routing space of the original design and prevents designers from doing iterative place and route. Limiting designers in this way results in performance degradation and possibly an

unroutable design. All of these requirements dramatically increase the cost of chip fabrication and time-to-market.

6.8 Summary

Experimental results with our fabricated malicious processor show that a new style of fabrication-time attack is possible; a fabrication-time attack that applies to a wide range of hardware, spans the digital and analog domains, and affords control to a remote attacker. Experimental results also show that A2 is effective at reducing the security of existing software, enabling unprivileged software full control over the processor. Finally, the experimental results demonstrate the elusive nature of A2: 1) A2 is as small as a single gate—two orders of magnitude smaller than a digital-only equivalent 2) attackers can add A2 to an existing circuit layout without perturbing the rest of the circuit 3) a diverse set of benchmarks fail to activate A2 and 4) A2 has little impact on circuit power, frequency, or delay.

Our results expose two weaknesses in current malicious hardware defenses. First, existing defenses analyze the digital behavior of a circuit using functional simulation or the analog behavior of a circuit using circuit simulation. Functional simulation is unable to capture the analog properties of an attack, while it is impractical to simulate an entire processor for thousands of clock cycles in a circuit simulator—this is why we had to fabricate A2 to verify that it worked. Second, the minimal impact on the run-time properties of a circuit (e.g., power, temperature, and delay) due to A2 suggests that it is an extremely challenging task for side-channel analysis techniques to detect this new class of attacks. We believe that our results motivate a different type of defense; a defense where trusted circuits monitor the execution of untrusted circuits, looking for out-of-specification behavior in the digital domain.

CHAPTER 7

Low-Power Temperature Sensor Using Exponential Sub-threshold Oscillation Dependence

7.1 Introduction

Thermal sensing is one of the most commonly desired features in IoT devices to monitor either environmental or system/chip conditions. Various types of CMOS temperature sensors have been proposed in literature. Most conventional temperature sensors are based on parasitic bipolar junction transistors (BJTs). These sensors measure temperature by comparing a temperature-dependent voltage to a temperature-insensitive reference voltage derived from base-emitter voltages of two BJTs biased at different collector currents (V_{BE} and ΔV_{BE}). The ratio between the PTAT and reference voltages is digitized by a high-resolution analog-to-digital converter (ADC). These sensors can achieve very high accuracy (down to $\pm 0.15^{\circ}C$ [64, 65]), but with power consumption in μW range, making them unsuitable for miniaturized battery-powered applications. Other sensing elements and sensor architectures are proposed for these applications. Common sensing elements include MOS current, Dynamic Threshold MOS (DTMOS) and resistors [66]. To get a digital output, most sensors employ voltage referenced ADCs or frequency referenced frequency-to-digital converters (FDC). ADC-based temperature sensors usually achieve better linearity and supply insensitivity, but surfers from larger power and area cost, as well as design complexity.

Therefore, to achieve optimal trade-off between performance (temperature accuracy, resolution, and supply insensitivity) and costs (power, area and design/testing costs) for given applica-

tions, both sensing element and sensor architecture requires further innovations. For example, most ultra-low-power temperature sensors are based on MOS current or resistors and therefore choose time-to-digital [67–69], or frequency-to-digital converters [70–72], because of their simpler design and lower power consumption. But these designs all sacrifice accuracy ($> 1°C$ inaccuracy after 2-point trimming) and noise-limited resolution compared to the combinations of BJT and ADC. More recently, a design using DTMOS to emulate BJT behavior and zoom-in ADC is introduced in [73]. This sensor achieves $\pm 0.4°C$ inaccuracy with 63mK resolution, while consuming $0.6\mu W$ power. However, the complexity and area of the zoom-in ADC design are not preferred.

In order to further optimize the sensor design and considering specific applications, we observe that almost all IoT systems incorporate a timing source (real-time clock, RTC) for time synchronization, data recording, and radio communication. Typically, these RTCs employ crystal or MEMS-based oscillators, or RC oscillators in extremely small form factor devices. We therefore propose to build a temperature sensor that uses a system's core RTC as a timing reference to minimize power/area overhead. The challenges here is that the timing reference is not perfect as assumed in previous FDC designs, RTC will have frequency drift over temperature and supply sensitivity. Therefore, the sensor design must be able to mitigate these impacts. For the sake of completeness, we also show a fully-integrated, standalone temperature sensor that includes an RC-based timing circuit for systems that might not have a timing source.

Timing-based temperature sensors generally use oscillators to perform the sensing or analog-to-digital conversion, which is simple but has poor accuracy and voltage sensitivity compared to voltage-referenced designs [66]. In this work, we use a sub-threshold oscillator as an exponential temperature-to-frequency converter, which reduces the impact of jitter of the sensing oscillator and reference timing source on sensor resolution, and also relaxes the requirement on the temperature stability of the timing reference. To accomplish this, we report: (1) an accurate fitting method derived from device models to transform the temperature dependence of sub-threshold current to a linear output after 2-point calibration; (2) a sub-threshold oscillator with stacked native NMOS header that achieves 1%/V (resulting in $0.13°C$/V) line sensitivity; and (3) a delay cell with excellent current-to-frequency linearity to further improve accuracy across process variations. Pushing more processing to the digital domain can benefit from technology scaling and is especially suitable for IoT applications where data are transmitted to a server for processing. The power/area over-

head of the temperature sensor (beyond the existing RTC) are 0.6nJ per conversion at 125S/s and 8865μm^2 in 180nm CMOS. Despite the low costs, the temperature sensor achieves -0.22/0.19°C inaccuracy (3σ value) with crystal oscillator and 0.76/0.76°C with a fully integrated RC oscillator, 73/90mK resolution, and 0.13/0.36(°C/V) voltage sensitivity for the sensors with crystal/RC references.

7.2 Temperature Sensing with Sub-threshold Current

In order to achieve high temperature sensitivity with low power consumption. The exponential temperature dependence of sub-threshold MOS device current is a natural candidate. However, previous sensor designs based on sub-threshold current all suffer from low accuracy and high line sensitivity. For example, the first sub-threshold temperature sensor described in [74] fits temperature output with a simple exponential function and therefore obtains 1.5 to 3.1°C inaccuracy. To improve the linearity of sub-threshold current-based temperature sensor, we derive a more accurate fitting model. Starting with MOS device sub-threshold current model in Equation 7.1, Equation 7.2 can be derived with α_1 and α_2 defined in Equation 7.3. Through transformation, it is found that $T \times ln(I_d)$ is linearly dependent on temperature and the two parameters α_1 and α_1 can be calculated from 2-point fitting for each manufactured sensor (Equation 7.4).

$$I_d = \mu(T_r)\left(\frac{T}{T_r}\right)^{-1.5} C_{ox}\frac{W}{L} exp\left(\frac{q(V_{gs}-V_{th})}{mkT}\right), V_{th} = V_{th0} - \kappa T \tag{7.1}$$

$$I_d = \alpha_1 T^{0.5} exp\left(\frac{\alpha_2}{T}\right) \tag{7.2}$$

$$\alpha_1 = \mu(T_r) C_{ox}\frac{W}{LT_r^{-1.5}}\left(\frac{k}{q}\right)^2 exp\left(\frac{q\kappa V_{th}}{mk}\right), \alpha_2 = \frac{q(V_{gs}-V_{th0})}{mk} \tag{7.3}$$

$$T \times ln(I_d) = ln(\alpha_1)T + 0.5 \times ln(T) + \alpha_2 \approx ln(\alpha_1)T + \alpha_2 \tag{7.4}$$

Since existing RTCs are expected to be used as timing reference, the simplest method to measure sub-threshold current is to build a sub-threshold oscillator and count it with fixed time inter-

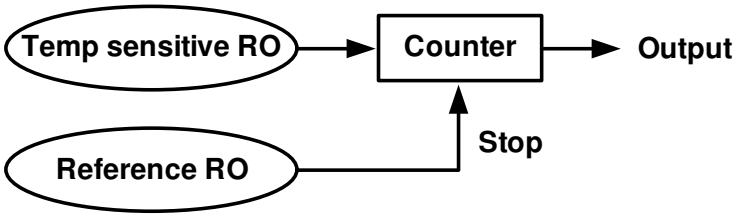


Figure 7.1: Basic temperature sensor design using oscillator-based sensing element.

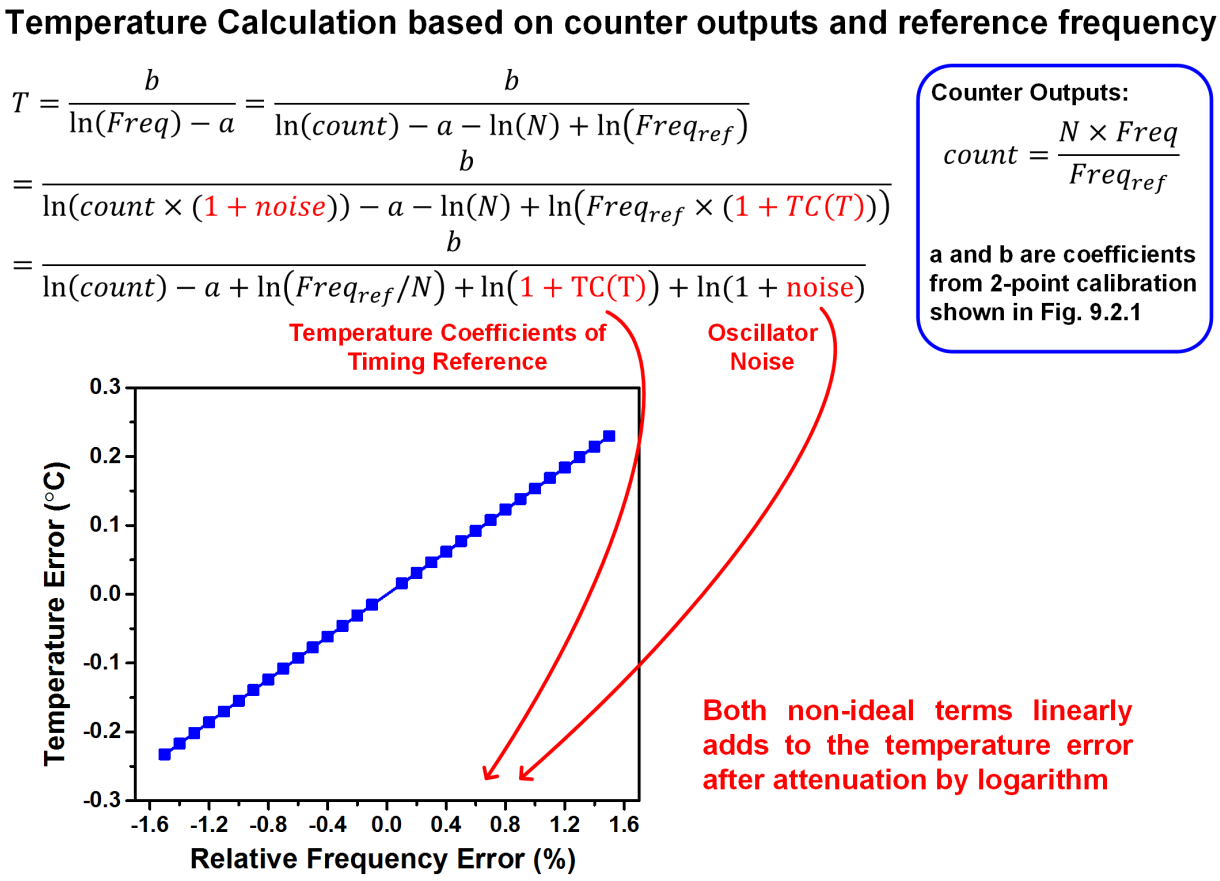


Figure 7.2: Benefits of exponential frequency dependence based on theoretical analysis and empirical results by adding frequency offsets to measurement results.

vals as shown in Figure 7.1. The sensing oscillator frequency can be decided by counter output and reference frequency. Using exponential temperature dependence brings another advantage here because it can significantly suppress the impacts of additional non-ideal noise and frequency shifts of the timing reference. These are linear terms added to the measured frequency and will be suppressed by logarithmic function as shown in Figure 7.2. A 1% frequency error will only result in $0.15^{\circ}C$ temperature error. As a result, it can achieve very good noise-limited resolution and high

accuracy, even with low-stability and noisy timing references.

7.3 Temperature Sensor Design Based on Sub-threshold Oscillator

The proposed sub-threshold oscillator is shown in Figure 7.3. A major challenge with using sub-threshold oscillators for temperature sensing is their high sensitivity to supply or bias voltage. To address this, we add a native NMOS (zero threshold voltage device) as a header device for the ring oscillator. This header behaves like a regulator with negative feedback. The regulated virtual VDD (VVDD) is determined by the V_{th} of the native header and the current drawn by the regulated oscillator. This voltage is around 300mV across temperature variations in our 180nm implementation. The oscillator current draw remains almost constant because of the staggered nature of oscillation and small on/off current ratio at deep sub-threshold region, which helps stabilize VVDD. Cascading two native NMOS headers further improves oscillator line sensitivity.

Pseudo-differential delay cells are used instead of single-ended inverters to further reduce fluctuation of current draw due to alternating switching of differential nets. Since we rely on the exponential temperature dependence of sub-threshold currents for temperature conversion, oscillator

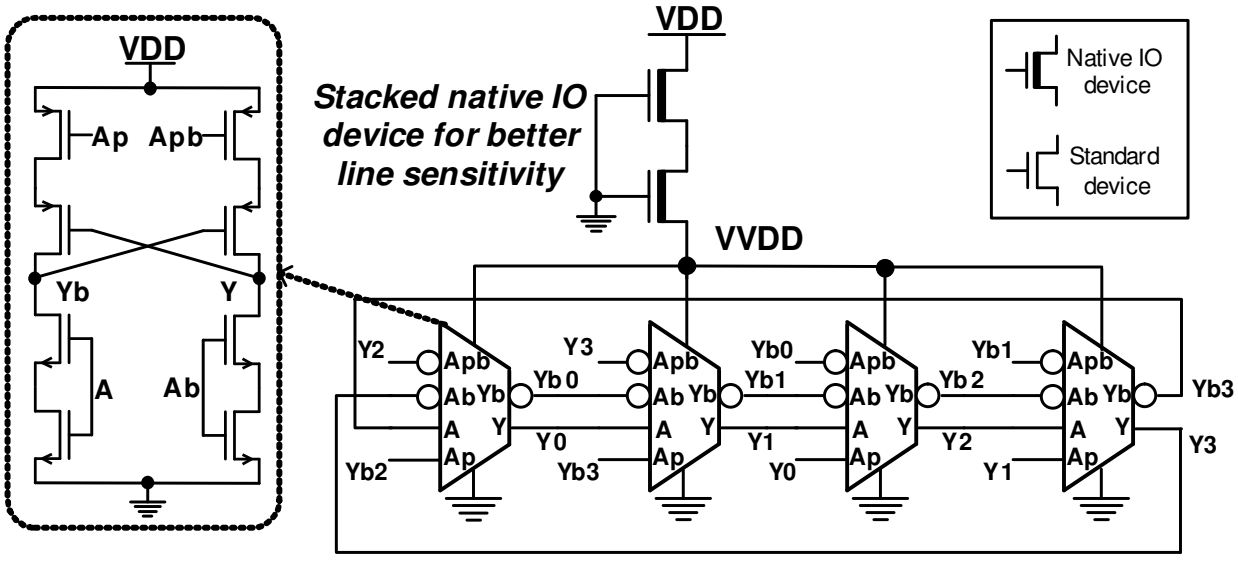


Figure 7.3: Working principle of temperature-sensing ring oscillator with native NMOS header for better line sensitivity.

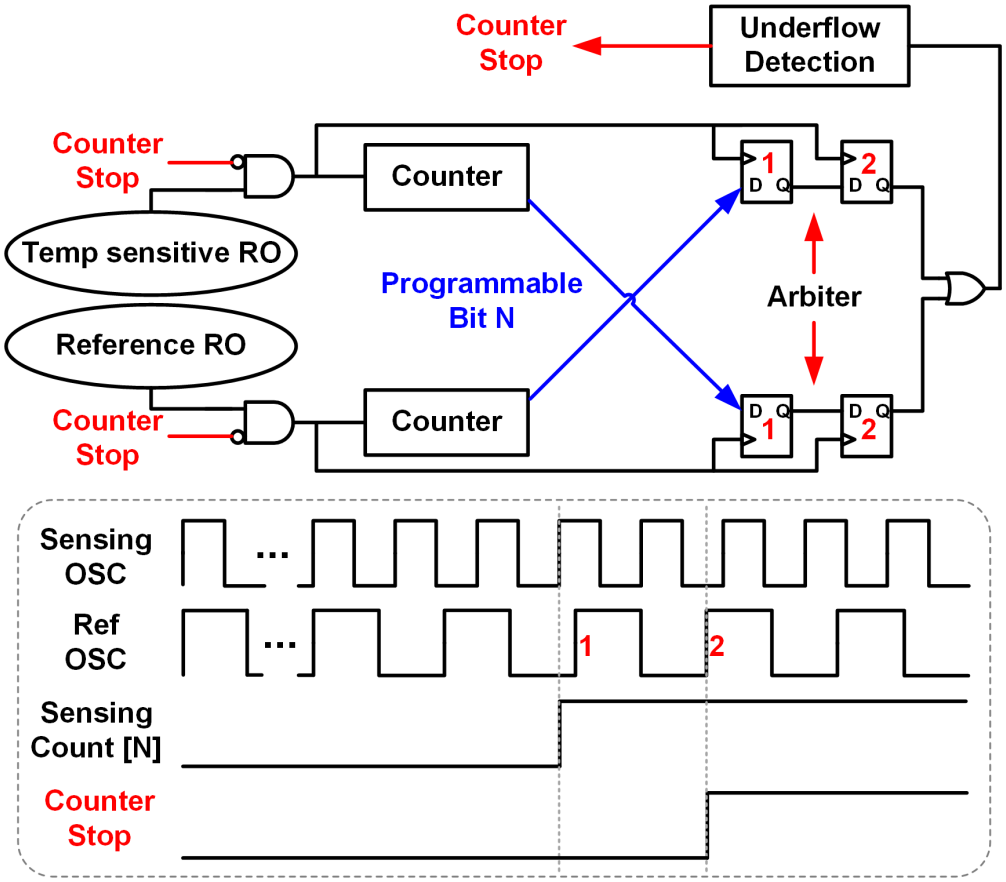


Figure 7.4: System diagram with sampling scheme to improve resolution, and exemplary waveform.

frequency should be linear with device currents. We develop a delay cell with PMOS switches controlled by preceding stages to eliminate the impact of rising/falling slopes on oscillator frequency, ensuring it is highly linear with sub-threshold current levels.

The temperature sensor consists of a sensing ring oscillator, a timing reference, and sampling circuits (Figure 7.4. Conventionally, frequency-to-digital converters are based on counting a sensing oscillator during a fixed number of reference cycles. This is not suitable for a sensing oscillator whose frequency depends exponentially on temperature, because its resolution and power consumption at high temperatures are orders of magnitude higher than that at lower temperature, resulting in unbalanced performance across temperature and poor energy efficiency. Moreover, the sensing oscillator speed can be either lower or higher than a typical RTC frequency (10s of kHz) as temperature varies. To better scale sensor performance across temperature, we propose a sampling scheme that limits the number of cycles of the faster oscillator (Figure 7.4). The circuit first finds

the faster oscillator to reach programmable 2^N cycles and then waits for two additional cycles of the slower oscillator before stopping both counters. Thus, conversion time is bounded by $2^N \times T_{ref}$ and the quantization resolution is decided by the faster oscillator rather than the slower one. Two additional cycles are implemented for synchronization metastability considerations. In addition, a counter underflow detector ensures a minimum of 2^n cycles on both counters before stopping them, ensuring enough conversion time.

7.4 Measurement Results

The temperature sensor is implemented in 180nm CMOS technology, which includes a sensing oscillator, counter and control logic. In order to evaluate the temperature sensor in applications that cannot afford a crystal or MEMS oscillator, a frequency-locking-based RC oscillator [75] is implemented on-chip to evaluate the temperature sensor under worst-case timing reference (50 to 100ppm/$^\circ C$ and 0.5%/V). The timing reference can also be supplied by a 32.768kHz crystal oscillator on testing board, which experiences the same temperature as the sensor. A die micrograph is provided in Figure 7.5.

To fully characterize the effects of process variation on accuracy of the temperature sensor, we measured both TT chips and skewed corner chips at all four corners. A total of 16 chips (8TT, 2FF,

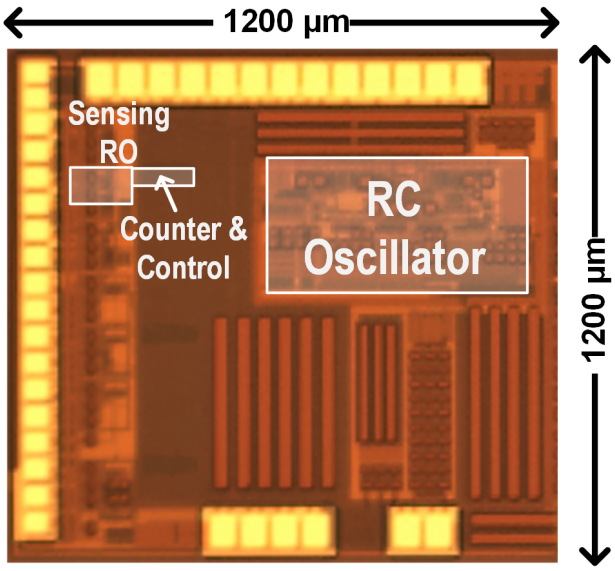
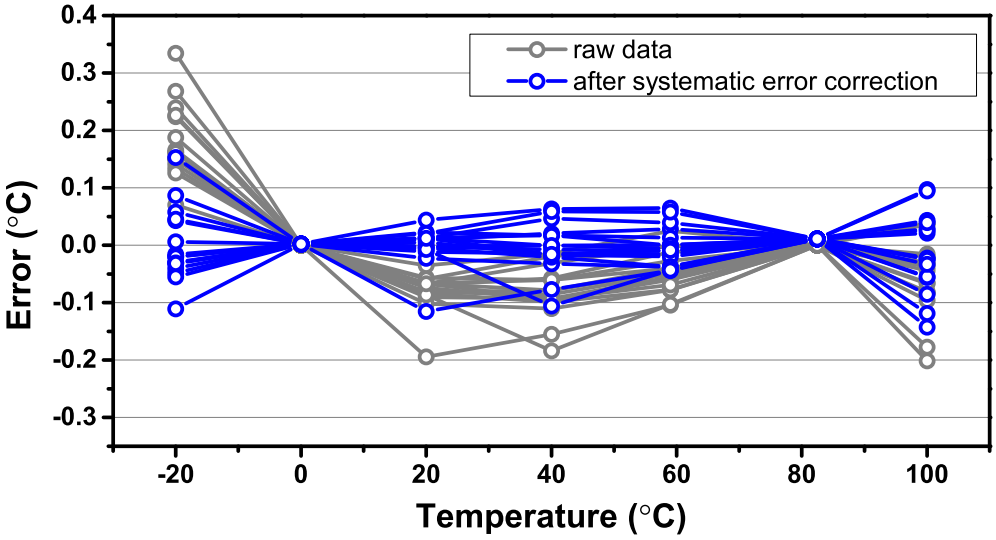
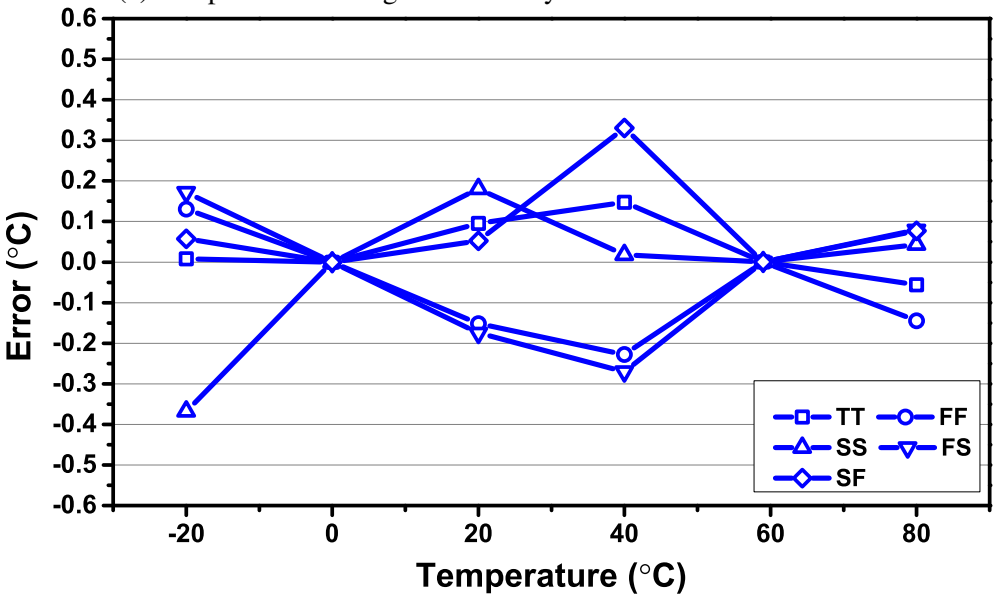


Figure 7.5: Die micrograph of 180nm temperature sensor.



(a) Temperature sensing core with crystal oscillator as reference.



(b) Fully integrated temperature sensor with on-chip RC oscillator as reference.

Figure 7.6: Temperature sensor inaccuracy after 2-point calibration.

2SS, 2FS, 2SF) are measured using the crystal oscillator frequency reference, showing excellent linearity of the sensing element. A single, universal systematic error correction, rather than lot-based correction, can be employed to improve the accuracy. Five integrated temperature sensors (one chip at each corner) are measured across temperature sweeps. As the on-chip resistor varies significantly at high temperatures, the operating range is reduced to 20 to 80°C, which is also the common operating range of RC oscillators. Figure 7.6 plots the inaccuracy results for both versions after 2-point calibration. It is also observed that the slope parameter in Equation 7.4 is

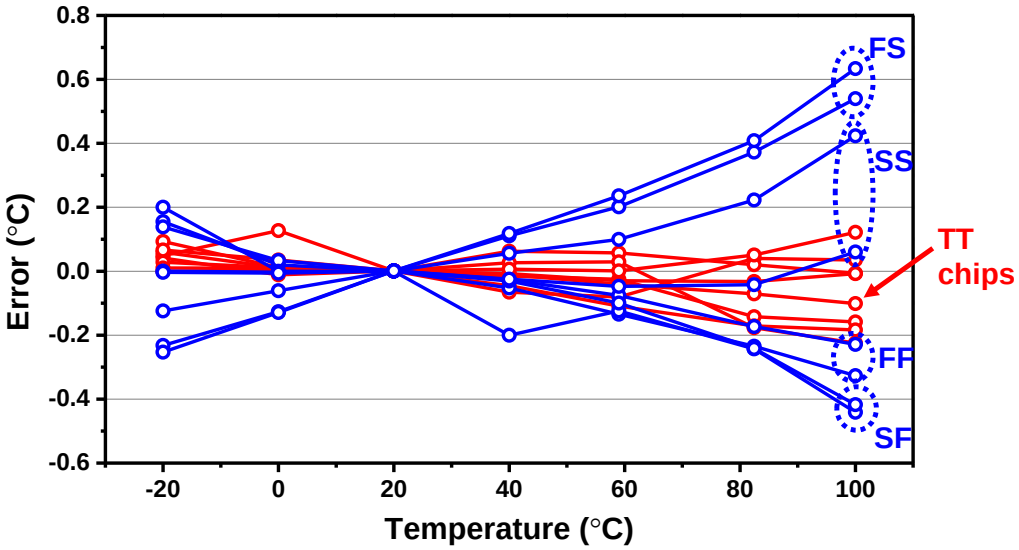


Figure 7.7: Inaccuracy of temperature sensor with crystal oscillator after 1-point calibration.

almost constant for sensors in same lot, which indicates that it is possible to do 1-point calibration at room temperature as shown in Figure 7.7. Even for different lots, using a constant value for the slope parameter only introduces about $\pm 0.6°C$ temperature measurement errors.

As discussed, the sensor suppresses noise significantly thanks to its exponential dependence on temperature, which is verified by measurements results shown in Figure 7.8. Since noise is significantly suppressed and the oscillators are running at low speed, the noise limited root-mean-square (RMS) resolution is close to quantization noise. As expected, the root-mean-square (RMS) resolution increases linearly with conversion time and reaches a noise floor after long enough conversion time. As can be seen 73 and 90mK RMS resolution are achieved with crystal and RC oscillator with a practical 8ms conversion time. However, the best resolution Figure-of-Merit (FoM) occurs at 1 and 2 seconds for 70kHz RC oscillator and 32.768kHz crystal oscillator, which are $0.48\text{pJ}\cdot K^2$ for sensing core only using crystal and $1.73\text{pJ}\cdot K^2$ for fully integrated version.

Temperature sensors for IoT devices may operate under fluctuating supply voltages since power management on these devices is often limited due to lack of good passive devices or low power budgets. The stacked native header significantly improves ring oscillator line sensitivity, which is around 1%/V shown in Figure 7.9. Considering the relation between measurement error and frequency offset shown in Figure 7.2, a voltage sensitivity of 0.13 and $0.36°C$/V is achieved with

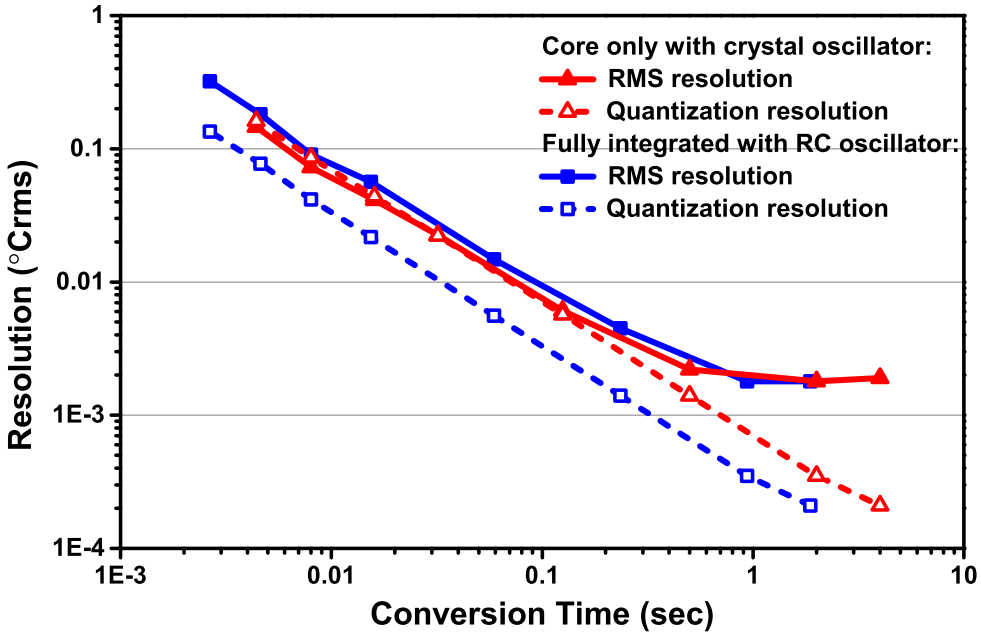


Figure 7.8: RMS and quantization resolution over conversion time.

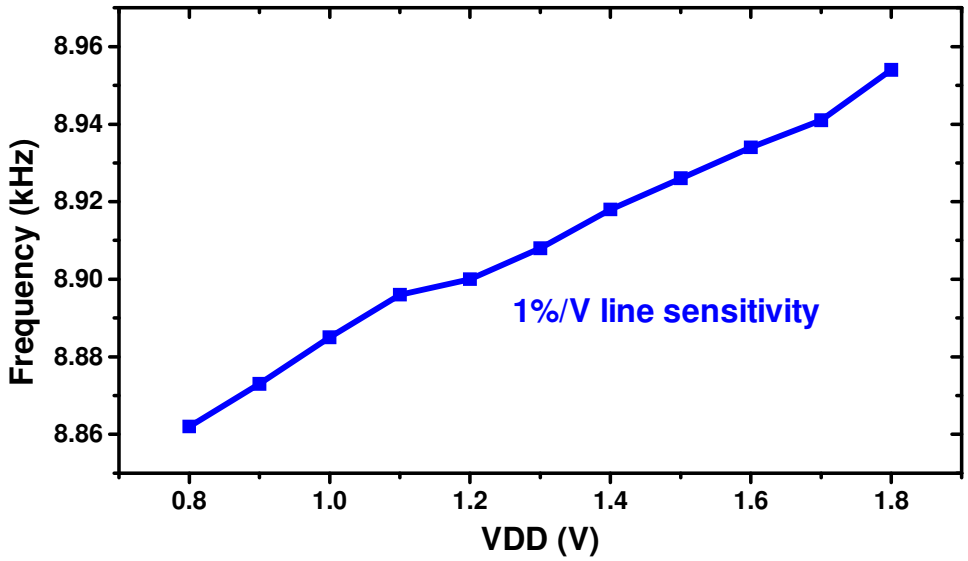


Figure 7.9: Line sensitivity of the sensing ring oscillator.

timing references using crystal oscillator and on-chip RC oscillator.

The 180nm temperature sensor core occupies only 8865μm^2 , which includes the sensing RO, counters, and state machine. The digital implementation makes the design very portable and highly scalable to new CMOS processes. The temperature sensor operates at 125 conversions/s while

Table 7.1: Summary of measurement results and comparison table with state-of-the-art MOS-based temperature sensors.

	This work		**VLSI 15 [69]**	**ISSCC 14 [73]**	**JSSC 14 [70]**	**JSSC 10 [68]**
	Core	**System**				
Technology	**180nm**		65nm	160nm	180nm	180nm
Type	**MOS**		MOS	DTMOS	MOS	MOS
Digital Conversion	**FDC**		FDC	ZSD2	FDC	TDC
Fully Integrated	**XO on board**	**Yes (RC Osc)**	Yes	No*	Yes	External clock
Area (μm^2)	**8865**	**220000**	4000	85000	90000	41600
Conversion Time (ms)	**8**	**8**	0.022	6	30	100
Power (µW)	**0.075**	**0.57**	154	0.6	0.071	0.12
Energy/Conversion (nJ)	**0.6**	**4.56**	3.388	3.6	2.13	3.6
Temperature Range (°C)	**-20~100**	**-20~80**	0~100	-40~125	0~100	-10~30
Inaccuracy (°C)	**-0.22/0.19 (2-point)**	**-0.76/0.76 ** (2-point)**	-0.9/0.9 (2-point)	-0.4/0.4 (1-point)	-1.4/1.5 (2-point)	-0.8/1 (2-point)
Resolution (mK)	**73**	**90**	300	63	300	200
Supply Voltage (V)	**1.2**	**1.2**	0.85~1.05	0.85	1.2	0.5
Voltage Sensitivity (°C/V)	**0.13**	**0.36**	34	0.45	14	Regulator
Resolution-FoM (pJ·K²)	**3.2**	**36.9**	304.92	14.29	191.7	144

* External OTA bias current and digital ADC backend.
** Estimated from only 5 chips at 5 corners, representing a pessimistic estimation.

consuming 0.6 and 4.56nJ per conversion at 1.2V for sensing core only and fully integrated temperature sensor. Figure 7.1 summarizes measurement results and compares to recent MOS-based low-power temperature sensors.

7.5 Summary

In conclusion, a tempearture sensor specifically designed for low-power Internet-of-Things systems. Existing real-time clocks in these systems are used as timing references to minimize overall power and area costs. The temperature sensor is all digital and can be ported to a wide range of technology nodes. The accuracy and resolution of the proposed temperature sensor is

close to much more complicated CMOS sensors using high-resolution delta-sigma analog to digital converters.

CHAPTER 8

Conclusions and Future Work

This work focuses on circuit techniques for low-power and secure Internet-of-Things systems, which are expected to significantly improve our daily lives and transform many industries. Hardware building blocks aiming at providing secure roots of trusts for the whole system are discussed in this work.

Chapter 2 and Chapter 3 introduce two true random number generators based on frequency collapse in multi-mode ring oscillators. The first one targets high robustness across wide temperature and voltage variations because IoT systems are likely to experience large temperature and internal voltage fluctuations. The second TRNG targets fully synthesizable design that can minimize design complexity and easy to be ported to different technologies. In addition to TRNGs, physically unclonable function is another emerging silicon security primitive with several potential applications including chip identification, secret key storage, and device authentication. Two types of PUFs exist in literature: "weak" PUF and "strong" PUF. Chapter 4 describes a 2-transistor amplifier-based "weak" PUF that breaks the trade-off between all desired metrics presented in previous works. A more powerful "Strong" PUF is presented in Chapter 5, which provides a runtime confidence level indicator to significantly improve PUF output reproducibility and provide more flexibility for PUF protocol design.

On the other side of security, Chapter 6 exploits the possibility of using analog behaviors of digital processors to construct a tiny Trojan that can be inserted directly into finished layout during fabrication time.

Lastly, the digital sub-threshold oscillator-based temperature sensor shown in Chapter 7 pro-

vides -0.22/0.19°C inaccuracy, 73mK resolution and 0.13°CV by making use of existing real time clocks in IoT systems.

The work presented in this thesis improves the performance trade-offs of several critical security and sensor blocks, which lays the foundation for emerging low-power and secure Internet-of-Things applications. At the same time, however, a few issues with security building blocks remain to be answered, especially the machine learning attacks targeting "strong" PUFs. At system level, in order to fully realize secure IoT systems, further optimizations involving circuit, architecture, network and algorithm are necessary.

BIBLIOGRAPHY

[1] G. Bell, "Bell's law for the birth and death of computer classes," *Commun. ACM*, vol. 51, no. 1, pp. 86–94, Jan. 2008. [Online]. Available: http://doi.acm.org/10.1145/1327452.1327453

[2] U. Today, "Hacked home devices caused massive Internet outage," 2016.

[3] N. News, "Jeep Hackers' are back with a scary new trick," 2016.

[4] CNN, "Cheney's defibrillator was modified to prevent hacking," 2013.

[5] National Institute of Standards and Technology, "A Statistical Test Suite for the Validation of Random Number Generators and Pseudo Random Number Generators for Cryptographic Applications," 2010.

[6] K. Yang, D. Blaauw, and D. Sylvester, "A robust 40 to 120°C all-digital true random number generator in 40nm CMOS," in *2015 Symposium on VLSI Circuits (VLSI Circuits)*, June 2015, pp. C248–C249.

[7] C. S. Petrie and J. A. Connelly, "A noise-based IC random number generator for applications in cryptography," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 47, no. 5, pp. 615–621, May 2000.

[8] R. Brederlow, R. Prakash, C. Paulus, and R. Thewes, "A low-power true random number generator using random telegraph noise of single oxide-traps," in *2006 IEEE International Solid State Circuits Conference - Digest of Technical Papers*, Feb 2006, pp. 1666–1675.

[9] M. Matsumoto, S. Yasuda, R. Ohba, K. Ikegami, T. Tanamoto, and S. Fujita, "1200 $\mu m^2$2 Physical Random-Number Generators Based on SiN MOSFET for Secure Smart-Card Application," in *2008 IEEE International Solid-State Circuits Conference - Digest of Technical Papers*, Feb 2008, pp. 414–624.

[10] S. K. Mathew, S. Srinivasan, M. A. Anders, H. Kaul, S. K. Hsu, F. Sheikh, A. Agarwal, S. Satpathy, and R. K. Krishnamurthy, "2.4 Gbps, 7 mW All-Digital PVT-Variation Tolerant True Random Number Generator for 45 nm CMOS High-Performance Microprocessors," *IEEE Journal of Solid-State Circuits*, vol. 47, no. 11, pp. 2807–2821, Nov 2012.

[11] C. Tokunaga, D. Blaauw, and T. Mudge, "True Random Number Generator With a Metastability-Based Quality Control," *IEEE Journal of Solid-State Circuits*, vol. 43, no. 1, pp. 78–85, Jan 2008.

[12] M. Bucci, L. Germani, R. Luzzi, A. Trifiletti, and M. Varanonuovo, "A high-speed oscillator-based truly random number source for cryptographic applications on a smart card IC," *IEEE Transactions on Computers*, vol. 52, no. 4, pp. 403–409, April 2003.

[13] B. Sunar, W. J. Martin, and D. R. Stinson, "A Provably Secure True Random Number Generator with Built-In Tolerance to Active Attacks," *IEEE Transactions on Computers*, vol. 56, no. 1, pp. 109–119, Jan 2007.

[14] K. Yang, D. Fick, M. B. Henry, Y. Lee, D. Blaauw, and D. Sylvester, "A 23Mb/s 23pJ/b fully synthesized true-random-number generator in 28nm and 65nm CMOS," in *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, Feb 2014, pp. 280–281.

[15] Q. Tang, B. Kim, Y. Lao, K. K. Parhi, and C. H. Kim, "True Random Number Generator circuits based on single- and multi-phase beat frequency detection," in *Proceedings of the IEEE 2014 Custom Integrated Circuits Conference*, Sept 2014, pp. 1–4.

[16] T. Amaki, M. Hashimoto, and T. Onoye, "A process and temperature tolerant oscillator-based true random number generator with dynamic 0/1 bias correction," in *2013 IEEE Asian Solid-State Circuits Conference (A-SSCC)*, Nov 2013, pp. 133–136.

[17] N. Liu, N. Pinckney, S. Hanson, D. Sylvester, and D. Blaauw, "A true random number generator using time-dependent dielectric breakdown," in *2011 Symposium on VLSI Circuits - Digest of Technical Papers*, June 2011, pp. 216–217.

[18] A. T. Markettos and S. W. Moore, "The Frequency Injection Attack on Ring-Oscillator-Based True Random Number Generators," in *Proceedings of the 11th International Workshop on Cryptographic Hardware and Embedded Systems*, ser. CHES '09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 317–331.

[19] M. Dichtl and J. D. Golić, *"High-Speed True Random Number Generation with Logic Gates Only"*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 45–62.

[20] A. A. Abidi, "Phase Noise and Jitter in CMOS Ring Oscillators," *IEEE Journal of Solid-State Circuits*, vol. 41, no. 8, pp. 1803–1816, Aug 2006.

[21] J. L. Folks and R. S. Chhikara, "The Inverse Gaussian Distribution and Its Statistical Application–A Review," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 40, no. 3, pp. 263–289, 1978. [Online]. Available: http://www.jstor.org/stable/2984691

[22] K. Yang, Q. Dong, D. Blaauw, and D. Sylvester, "14.2 A physically unclonable function with BER $< 10^{-8}$for robust chip authentication using oscillator collapse in 40nm CMOS," in *2015 IEEE International Solid-State Circuits Conference - (ISSCC) Digest of Technical Papers*, Feb 2015, pp. 1–3.

[23] J. W. Lee, D. Lim, B. Gassend, G. E. Suh, M. van Dijk, and S. Devadas, "A technique to build a secret key in integrated circuits for identification and authentication applications," in *2004 Symposium on VLSI Circuits. Digest of Technical Papers (IEEE Cat. No.04CH37525)*, June 2004, pp. 176–179.

[24] B. Karpinskyy, Y. Lee, Y. Choi, Y. Kim, M. Noh, and S. Lee, "8.7 Physically unclonable function for secure key generation with a key error rate of $2E-38$ in 45nm smart-card chips," in *2016 IEEE International Solid-State Circuits Conference (ISSCC)*, Jan 2016, pp. 158–160.

[25] A. Alvarez, W. Zhao, and M. Alioto, "14.3 15fJ/b static physically unclonable functions for secure chip identification with $< 2\%$ native bit instability and $140\times$ Inter/Intra PUF hamming

distance separation in 65nm," in *2015 IEEE International Solid-State Circuits Conference - (ISSCC) Digest of Technical Papers*, Feb 2015, pp. 1–3.

[26] J. Li and M. Seok, "A 3.07 μm^2/bitcell physically unclonable function with 3.5% and 1% bit-instability across 0 to 80°*C*C and 0.6 to 1.2V in a 65nm CMOS," in *2015 Symposium on VLSI Circuits (VLSI Circuits)*, June 2015, pp. C250–C251.

[27] S. K. Mathew, S. K. Satpathy, M. A. Anders, H. Kaul, S. K. Hsu, A. Agarwal, G. K. Chen, R. J. Parker, R. K. Krishnamurthy, and V. De, "16.2 A 0.19pJ/b PVT-variation-tolerant hybrid physically unclonable function circuit for 100% stable secure key generation in 22nm CMOS," in *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, Feb 2014, pp. 278–279.

[28] Y. Su, J. Holleman, and B. P. Otis, "A Digital 1.6 pJ/bit Chip Identification Circuit Using Process Variations," *IEEE Journal of Solid-State Circuits*, vol. 43, no. 1, pp. 69–77, Jan 2008.

[29] G. E. Suh and S. Devadas, "Physical Unclonable Functions for Device Authentication and Secret Key Generation," in *2007 44th ACM/IEEE Design Automation Conference*, June 2007, pp. 9–14.

[30] S. Stanzione, D. Puntin, and G. Iannaccone, "CMOS Silicon Physical Unclonable Functions Based on Intrinsic Process Variability," *IEEE Journal of Solid-State Circuits*, vol. 46, no. 6, pp. 1456–1463, June 2011.

[31] N. Liu, S. Hanson, D. Sylvester, and D. Blaauw, "OxID: On-chip one-time random ID generation using oxide breakdown," in *2010 Symposium on VLSI Circuits*, June 2010, pp. 231–232.

[32] M.-L. Li, P. Ramachandran, S. K. Sahoo, S. V. Adve, V. S. Adve, and Y. Zhou, "Understanding the Propagation of Hard Errors to Software and Implications for Resilient System Design," in *International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS. Seattle, WA: ACM, Mar. 2008, pp. 265–276.

[33] M. Hicks, C. Sturton, S. T. King, and J. M. Smith, "Specs: A lightweight runtime mechanism for protecting software from security-critical processor bugs," in *Proceedings of the Twentieth International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS. Istanbul, Turkey: ACM, 2015, pp. 517–529.

[34] S. S. Technology. (2012, Oct.) Why node shrinks are no longer offsetting equipment costs. [Online]. Available: http://electroiq.com/blog/2012/10/why-node-shrinks-are-no-longer-offsetting-equipment-costs/

[35] G. T. Becker, F. Regazzoni, C. Paar, and W. P. Burleson, "Stealthy Dopant-level Hardware Trojans," in *International Conference on Cryptographic Hardware and Embedded Systems*, ser. CHES. Berlin, Heidelberg: Springer-Verlag, 2013, pp. 197–214.

[36] R. Kumar, P. Jovanovic, W. Burleson, and I. Polian, "Parametric Trojans for Fault-Injection Attacks on Cryptographic Hardware," in *Workshop on Fault Diagnosis and Tolerance in Cryptography*, ser. FDT, 2014, pp. 18–28.

[37] T. Sugawara, D. Suzuki, R. Fujii, S. Tawa, R. Hori, M. Shiozaki, and T. Fujino, "Reversing Stealthy Dopant-Level Circuits," in *International Conference on Cryptographic Hardware and Embedded Systems*, ser. CHES. New York, NY: Springer-Verlag, 2014, pp. 112–126.

[38] D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi, and B. Sunar, "Trojan Detection Using IC Fingerprinting," in *Symposium on Security and Privacy*, ser. S&P. Washington, DC: IEEE Computer Society, 2007, pp. 296–310.

[39] Y. Jin and Y. Makris, "Hardware Trojan Detection Using Path Delay Fingerprint," in *Hardware-Oriented Security and Trust*, ser. HOST. Washington, DC: IEEE Computer Society, 2008, pp. 51–57. [Online]. Available: http://dx.doi.org/10.1109/HST.2008.4559049

[40] M. Potkonjak, A. Nahapetian, M. Nelson, and T. Massey, "Hardware Trojan horse detection using gate-level characterization," in *Design Automation Conference*, ser. DAC, vol. 46, 2009, pp. 688–693.

[41] S. Narasimhan, X. Wang, D. Du, R. S. Chakraborty, and S. Bhunia, "TeSR: A Robust Temporal Self-Referencing Approach for Hardware Trojan Detection," in *Hardware-Oriented Security and Trust*, ser. HOST. San Diego, CA: IEEE Computer Society, Jun. 2011, pp. 71–74.

[42] J. Li and J. Lach, "At-speed Delay Characterization for IC Authentication and Trojan Horse Detection," in *Hardware-Oriented Security and Trust*, ser. HOST. Washington, DC: IEEE Computer Society, 2008, pp. 8–14.

[43] D. Forte, C. Bao, and A. Srivastava, "Temperature Tracking: An Innovative Run-time Approach for Hardware Trojan Detection," in *International Conference on Computer-Aided Design*, ser. ICCAD. IEEE, 2013, pp. 532–539.

[44] S. Kelly, X. Zhang, M. Tehranipoor, and A. Ferraiuolo, "Detecting Hardware Trojans Using On-chip Sensors in an ASIC Design," *Journal of Electronic Testing*, vol. 31, no. 1, pp. 11–26, Feb. 2015. [Online]. Available: http://dx.doi.org/10.1007/s10836-015-5504-x

[45] A. Waksman and S. Sethumadhavan, "Silencing Hardware Backdoors," in *IEEE Security and Privacy*, ser. S&P. Oakland, CA: IEEE Computer Society, May 2011.

[46] X. Wang, S. Narasimhan, A. Krishna, T. Mal-Sarkar, and S. Bhunia, "Sequential hardware trojan: Side-channel aware design and placement," in *Computer Design (ICCD), 2011 IEEE 29th International Conference on*, Oct 2011, pp. 297–300.

[47] OpenCores.org. OpenRISC OR1200 processor. [Online]. Available: http://opencores.org/or1k/OR1200_OpenRISC_Processor

[48] M. R. Guthaus, J. S. Ringenberg, D. Ernst, T. M. Austin, T. Mudge, and R. B. Brown, "MiBench: A free, commercially representative embedded benchmark suite," in *Workshop on Workload Characterization*. Washington, DC: IEEE Computer Society, 2001, pp. 3–14.

[49] K. Yang, M. Hicks, Q. Dong, T. Austin, and D. Sylvester, "A2: Analog malicious hardware," https://github.com/impedimentToProgress/A2, 2016.

[50] M. Rostami, F. Koushanfar, J. Rajendran, and R. Karri, "Hardware security: Threat models and metrics," in *Proceedings of the International Conference on Computer-Aided Design*, ser. ICCAD '13. San Jose, CA: IEEE Press, 2013, pp. 819–823.

[51] M. Hicks, M. Finnicum, S. T. King, M. M. K. Martin, and J. M. Smith, "Overcoming an Untrusted Computing Base: Detecting and Removing Malicious Hardware Automatically," *USENIX ;login*, vol. 35, no. 6, pp. 31–41, Dec. 2010. [Online]. Available: https://www.usenix.org/system/files/login/articles/hicks.pdf

[52] S. T. King, J. Tucek, A. Cozzie, C. Grier, W. n. Jiang, and Y. Zhou, "Designing and implementing malicious hardware," in *Workshop on Large-Scale Exploits and Emergent Threats*, ser. LEET, vol. 1, Apr. 2008.

[53] J. Rajendran, O. Sinanoglu, and R. Karri, "Is split manufacturing secure?" in *Design, Automation and Test in Europe*, ser. DATE, 2013, pp. 1259–1264.

[54] K. Vaidyanathan, B. Das, and L. Pileggi, "Detecting reliability attacks during split fabrication using test-only BEOL stack," in *Design Automation Conference*, ser. DAC, vol. 51. IEEE, Jun. 2014, pp. 1–6.

[55] K. Vaidyanathan, B. Das, E. Sumbul, R. Liu, and L. Pileggi, "Building trusted ICs using split fabrication," in *International Symposium on Hardware-Oriented Security and Trust*, ser. HOST. IEEE Computer Society, 2014, pp. 1–6.

[56] K. Vaidyanathan, R. Liu, E. Sumbul, Q. Zhu, F. Franchetti, and L. Pileggi, "Efficient and secure intellectual property (IP) design with split fabrication," in *International Symposium on Hardware-Oriented Security and Trust*, ser. HOST. IEEE Computer Society, 2014, pp. 13–18.

[57] F. Imeson, A. Emtenan, S. Garg, and M. V. Tripunitara, "Securing Computer Hardware Using 3d Integrated Circuit (IC) Technology and Split Manufacturing for Obfuscation," in *Conference on Security*, ser. Security. USENIX Association, 2013, pp. 495–510. [Online]. Available: http://dl.acm.org/citation.cfm?id=2534766.2534809

[58] L. Lin, M. Kasper, T. Gneysu, C. Paar, and W. Burleson, "Trojan Side-Channels: Lightweight Hardware Trojans Through Side-Channel Engineering," in *International Workshop on Cryptographic Hardware and Embedded Systems*, ser. CHES, vol. 11. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 382–395.

[59] Y. Shiyanovskii, F. Wolff, A. Rajendran, C. Papachristou, D. Weyer, and W. Clay, "Process reliability based trojans through NBTI and HCI effects," in *Conference on Adaptive Hardware and Systems*, ser. AHS, 2010, pp. 215–222.

[60] A. Pellegrini, V. Bertacco, and T. Austin, "Fault-based attack of rsa authentication," in *Design, Automation Test in Europe Conference Exhibition (DATE), 2010*, March 2010, pp. 855–860.

[61] J. Balasch, B. Gierlichs, and I. Verbauwhede, "Electromagnetic circuit fingerprints for hardware trojan detection," in *Electromagnetic Compatibility (EMC), 2015 IEEE International Symposium on*, Aug 2015, pp. 246–251.

[62] D. Ernst, N. S. Kim, S. Das, S. Pant, R. Rao, T. Pham, C. Ziesler, D. Blaauw, T. Austin, K. Flautner, and T. Mudge, "Razor: a low-power pipeline based on circuit-level timing speculation," in *Microarchitecture, 2003. MICRO-36. Proceedings. 36th Annual IEEE/ACM International Symposium on*, Dec 2003, pp. 7–18.

[63] K. Xiao, D. Forte, and M. Tehranipoor, "A novel built-in self-authentication technique to prevent inserting hardware trojans," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 33, no. 12, pp. 1778–1791, Dec 2014.

[64] K. Souri, Y. Chae, and K. A. A. Makinwa, "A CMOS Temperature Sensor With a Voltage-Calibrated Inaccuracy of 0.15 C (3) From 55 C to 125 C," *IEEE Journal of Solid-State Circuits*, vol. 48, no. 1, pp. 292–301, Jan. 2013.

[65] K. Souri and K. A. A. Makinwa, "A 0.12 mm 7.4 W Micropower Temperature Sensor With an Inaccuracy of 0.2 C (3) From 30 C to 125 C," *IEEE Journal of Solid-State Circuits*, vol. 46, no. 7, pp. 1693–1700, Jul. 2011.

[66] K. Makinwa. (2016, Jun) Smart temperature sensor survey. [Online]. Available: http://ei.ewi.tudelft.nl/docs/TSensor_survey.xls

[67] M. K. Law and A. Bermak, "A 405-nW CMOS Temperature Sensor Based on Linear MOS Operation," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 56, no. 12, pp. 891–895, Dec 2009.

[68] M. K. Law, A. Bermak, and H. C. Luong, "A sub-*mu* w embedded cmos temperature sensor for rfid food monitoring application," *IEEE Journal of Solid-State Circuits*, vol. 45, no. 6, pp. 1246–1255, June 2010.

[69] T. Anand, K. A. A. Makinwa, and P. K. Hanumolu, "A self-referenced VCO-based temperature sensor with 0.034°*C*/mV supply sensitivity in 65nm CMOS," in *2015 Symposium on VLSI Circuits (VLSI Circuits)*, June 2015, pp. C200–C201.

[70] S. Jeong, Z. Foo, Y. Lee, J. Y. Sim, D. Blaauw, and D. Sylvester, "A Fully-Integrated 71nW CMOS Temperature Sensor for Low Power Wireless Sensor Nodes," *IEEE Journal of Solid-State Circuits*, vol. 49, no. 8, pp. 1682–1693, Aug 2014.

[71] A. Vaz, A. Ubarretxena, I. Zalbide, D. Pardo, H. Solar, A. Garcia-Alonso, and R. Berenguer, "Full passive uhf tag with a temperature sensor suitable for human body temperature monitoring," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 57, no. 2, pp. 95–99, Feb 2010.

[72] Z. Shenghua and W. Nanjian, "A novel ultra low power temperature sensor for uhf rfid tag chip," in *2007 IEEE Asian Solid-State Circuits Conference*, Nov 2007, pp. 464–467.

[73] K. Souri, Y. Chae, F. Thus, and K. Makinwa, "A 0.85V 600nW all-CMOS temperature sensor with an inaccuracy of $\pm 0.4^{\circ}C(3\sigma)$ from -40 to $125^{\circ}C$," in *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, Feb 2014, pp. 222–223.

[74] E. Saneyoshi, K. Nose, M. Kajita, and M. Mizuno, "A 1.1V $35\mu m \times 35\mu m$ thermal sensor with supply voltage sensitivity of $2^{\circ}C$/10%-supply for thermal management on the SX-9 supercomputer," in *2008 IEEE Symposium on VLSI Circuits*, June 2008, pp. 152–153.

[75] M. Choi, S. Bang, T. K. Jang, D. Blaauw, and D. Sylvester, "A 99nW 70.4kHz resistive frequency locking on-chip oscillator with 27.4ppm/$^{\circ}C$ temperature stability," in *2015 Symposium on VLSI Circuits (VLSI Circuits)*, June 2015, pp. C238–C239.